UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

FACULTAD DE INGENIERÍA ELÉCTRICA
DIVISIÓN DE ESTUDIOS DE POSGRADO

"DETERMINACIÓN DEL ESTADO ESTACIONARIO PERIÓDICO DE SISTEMAS ELÉCTRICOS NO-LINEALES MEDIANTE UN MÉTODO NEWTON BASADO EN LA MATRIZ EXPONENCIAL DISCRETA, INCORPORANDO TÉCNICAS DE DISPERSIDAD Y PROCESAMIENTO EN PARALELO"

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

PRESENTA

FÉLIX GUADALUPE ORTEGA RAMÍREZ

DIRECTOR DE TESIS

DR. J. AURELIO MEDINA RÍOS

CO-DIRECTOR DE TESIS

DR. JUAN SEGUNDO RAMÍREZ

MORELIA, MICHOACÁN FEBRERO DE 2011

Dedicatoria

A los seres que más amo en este mundo: A mi padre Félix Ortega Camargo, mi madre Laura Ramírez Santarrosa y a mis hermanas, Teresita, Laura Paulina y María Guadalupe, en quienes han sido la fuente de mi inspiración y encontrado siempre un apoyo incondicional.

Agradecimientos

Deseo agradecer profundamente a la Universidad Michoacana de San Nicolás de Hidalgo (UMSNH), ya que me brindo la oportunidad de cursar la licenciatura y maestría en sus aulas. Así como al Consejo Nacional de Ciencia y Tecnología (CONACyT), por el apoyo económico durante el transcurso de la maestría.

A todos los profesores de este posgrado, por transmitir sus conocimientos y siempre dar un consejo cuando fue necesario.

Sin lugar a duda tengo que expresar mi más sincero agradecimiento a mi asesor de tesis el Dr. Aurelio Medina Ríos por compartir sus bastos conocimientos del tema y guiarme en la realización de este trabajo.

Como olvidar al Dr. Antonio Ramos Paz y el Dr. Juan Segundo debo expresarles mi gratitud; por tener la paciencia ante mis dudas de novato, por escuchar atentamente los problemas que surgieron a lo largo de esta tesis y ayudarme a resolverlos.

Deseo agradecer profundamente a mis padres y hermanas por el apoyo brindado durante mi vida, seria difícil imaginar el andar diario, sin su comprensión, su apoyo inmenso y su amor. ¡Gracias! A mis padres por compartir y dedicar gran parte de sus vidas conmigo y darme aliento para seguir adelante en mis estudios. Quiero enfatizar mi agradecimiento a mi padre y madre por brindarme sabios consejos en los momentos difíciles de mi vida.

A mis amigos que siempre me brindaron su apoyo en los momentos mas difíciles y con quienes pasé agradables momentos y que me brindaron su apoyo en los momentos más difíciles, especialmente a Víctor Armando, Omar Rico, José Roberto, Julio Cesar, Omar Beltrán, José Pilar, Mario Leal y Gustavo Hernández.

Y a todas las personas que contribuyeron directa o indirectamente para poder culminar esta tesis.

Resumen

Esta tesis describe la aplicación de técnicas de dispersidad y procesamiento en paralelo a la técnica Newton basada en la Matriz Exponencial Discreta que permite acelerar la convergencia de las variables de estado al Ciclo Límite para obtener la solución en estado estacionario periódico de redes eléctricas con componentes lineales y no lineales.

La eficiencia de este proceso puede ser mejorado de forma significativa con la aplicación de técnicas de dispersidad, para lo cual se utiliza la librería CSparse. La plataforma que se utiliza para realizar el procesamiento en paralelo en este trabajo de investigación es OpenMP.

En esta tesis se presentan comparaciones de las convergencias que tienen los métodos Newton de aceleración de la convergencia al Ciclo Límite basados en procedimientos de Diferenciación Numérica y Matriz Exponencial Discreta, respectivamente; se presentan las eficiencias obtenidas con la incorporación de procesamiento en paralelo. También se describe el esfuerzo computacional asociado con la solución computacional obtenida utilizando un método convencional de solución de Fuerza Bruta.

Abstract

This thesis describes the application of sparsity techniques and parallel processing to the Newton technique based on the Discrete Exponential Matrix, which allows to accelerate the convergence of the state variables to the Limit Cycle to obtain the periodic steady state solution of a power networks with linear, nonlinear and time-varying components steady.

This process can be dramatically improved with the application of sparsity. To achieve this goal the library CSparse was used. In this research, the OpenMP parallel processing platform was used.

The Newton methods based of Numerical Diffrentiation and Discrete Exponential Matrix procedures are compared in terms of convergence to the limit cycle. Additionally, computational efficiency is shown when parallel processing is used. The computational effort associated with the periodic steady state solution of nonlinear power networks using a conventional (Brute Force) method is described as well.

Índice general

Índice de Figuras

Índice de Tablas

Lista de Símbolos y Abreviaturas

ε Perturbación para el cálculo numérico de Jacobiano

ξ Criterio de convergencia

E Eficiencia

$E_{relativa}$ Eficiencia relativa

p Elementos de proceso

$p.u.$ Por unidad

AC Corriente Alterna

AD Aproximación Directa

C Capacitancia

DC Corriente Directa

DN Diferenciación Numérica

EDO's Ecuaciones Diferenciales Ordinarias

EEP Estado Estacionario Periódico

Exp Exponente

FB Fuerza Bruta

L Inductancia

MED Matriz Exponencial Discreta

R Resistencia

RK4 Runge-Kutta de cuarto orden

Capítulo 1

Introducción

La solución en Estado Estacionario Periódico (EEP) de sistemas eléctricos, se puede obtener mediante metodologías en el dominio del tiempo, de la frecuencia e híbridos frecuencia-tiempo [Kundert *et al*. 1990]. En particular, en esta tesis el estado estacionario periódico de sistemas eléctricos no lineales se obtiene en el dominio del tiempo basado en la aplicación del mapa de Poincaré, extrapolación al ciclo limite y un método Newton basado en la determinación de la Matriz Exponencial Discreta (MED) [Segundo y Medina 2010a]. Una de las alternativas de solución de los métodos Newton, está basado en la Matriz Exponencial Discreta [Segundo y Medina 2010a], en la cual se producen matrices altamente dispersas durante el proceso de solución, puede resultar computacionalmente demandante. Este proceso puede ser mejorado de forma significativa con la incorporación de técnicas de dispersidad [Timothy 2006].

El proceso de solución en que está basado el método MED permite la solución en cada paso de integración del sistema de ecuaciones, en particular, en el proceso de identificación de la matriz de transición de estados que requiere el método Newton de solución, lo que hace ideal y propicia la aplicación de técnicas de procesamiento en paralelo. El procesamiento en paralelo se realiza utilizando una computadora Intel (R) Xeon (R) CPU E5405 con dos procesadores de 4 núcleos cada uno y una velocidad de 2.0 GHz y programación basada en OpenMP.

1.1. Antecedentes

Hoy en día para el diseño de circuitos electrónicos, así como para cuestiones de calidad de la energía eléctrica, se necesita simular las redes eléctricas de manera que se pueda obtener su comportamiento en estado estacionario periódico. Para llegar a esto se requiere la aplicación de algoritmos de integración eficientes que nos lleven a la rápida solución en estado

estacionario.

Los cálculos involucrados en el análisis y simulación digital de los sistemas eléctricos de potencia bajo distintos escenarios de operación son cada día más complejos y requieren de una mayor capacidad de procesamiento; estos cálculos requieren ser realizados con la mayor rapidez y eficiencia posible, mediante el uso de herramientas avanzadas de cómputo, matemáticas y numéricas, con el propósito de lograr simulaciones eficientes y precisas de los sistemas eléctricos de potencia sujetos a diversos escenarios de operación, así como representar adecuadamente diversos tipos de dispositivos, elementos y topologías.

1.1.1. Técnicas para la determinación del estado estacionario periódico

Se han desarrollado e implementado distintas metodologías para determinar la solución en EEP en diferentes marcos de referencia y así obtener estudios precisos de análisis armónicos. En [Kundert *et al*. 1990] se comparan las características de los métodos y se pueden clasificar en las siguientes categorías:

- Métodos en el dominio del tiempo
- Métodos en el dominio de la frecuencia
- Métodos híbridos (Frecuencia-Tiempo)

Métodos en el dominio del tiempo

El modelo matemático que describe la dinámica de un sistema eléctrico de potencia es en general un conjunto no autónomo de ecuaciones diferenciales no lineales. Así que la solución de estado estacionario del sistema eléctrico puede obtenerse directamente integrando las Ecuaciones Diferenciales Ordinarias (EDO's) que modelan la dinámica del sistema durante un lapso de tiempo suficientemente grande como para que el transitorio pueda ser despreciable [Parker y Chua 1989]. Este procedimiento, conocido como método de Fuerza Bruta (FB), es el más simple [Parker y Chua 1989]. Sin embargo, tiene algunos inconvenientes que lo hacen en muchas ocasiones inapropiado, tal como en el caso de que el sistema en cuestión esté pobremente amortiguado o que sea difícil de integrar a una alta precisión, como es el caso de los sistemas rígidos. Para sortear los inconvenientes del método de FB , se han propuesto otros métodos basados en el método de Newton, que son conocidos como métodos de disparo [Parker y Chua 1989].

La primera contribución orientada a la determinación rápida del estado estacionario periódico de redes nolineales (electrónicas) en el dominio del tiempo es debida a Aprille y Trick

[Aprille y Trick 1972]. En otras contribuciones [Bedrosian y Vlach 1992, Donde y Hiskens 2006], los métodos de disparo han sido aplicados a sistemas eléctricos que contienen dispositivos de conmutación.

Alternativamente, a los métodos de disparo, se han propuesto métodos Newton basados en el mapa de Poincaré [Parker y Chua 1989] para el cálculo de la solución en EEP. En el trabajo presentado en [Semlyen y Medina 1995] se describen tres técnicas Newton que permiten la aceleración de las variables de estado al Ciclo Límite. Los métodos propuestos son: Diferenciación Numérica (DN), Aproximación Directa (AD) y Matriz Exponencial (ME), aunque se reportan únicamente los resultados de la aplicación de los métodos DN y AD.

En el trabajo [Garcia y Acha 2004] se presenta la aplicación de técnicas de estás técnicas en el análisis de redes eléctricas trifásicas para estudios de armónicos de calidad de la energía. En [Chang *et. al* 2006] se determina la solución en EEP de redes con convertidores de AC/DC, utilizando el mapa de Poincaré.

En [Segundo y Medina 2008] se detalla una técnica en el dominio del tiempo para la solución en EEP de sistemas eléctricos con controladores de flujo de potencia unificado (UPFC, por sus siglas en inglés) incorporando tres estrategias para la representación del proceso de conmutación del UPFC, la solución en EEP es obtenida con el método Newton basado en el proceso de DN. Recientemente en la contribución hecha en [Segundo y Medina 2010a] se propone una metodología eficiente para la solución EEP de sistemas eléctricos no lineales, que está basado en el método de la Matriz Exponencial Discreta (MED).

En [Lian y Noda 2010] se presenta un método de flujos de potencia armónicos, el cual se implementa completamente en el dominio del tiempo, el método propuesto esencialmente es una extensión del método de disparo para incluir las restricciones de flujos de potencia y así poder agregar cargas en los cálculos de flujos de potencia.

Métodos en el dominio de la frecuencia

Los métodos en el dominio de la frecuencia son muy utilizados debido a la facilidad que se tiene en la construcción del conjunto de ecuaciones que describen la dinámica de las redes eléctricas a ser analizadas. En [Medina *et al.* 2007] los métodos en el dominio de la frecuencia son divididos en las siguientes categorías: método directo, análisis armónico iterativo y flujos de potencia armónicos.

El método directo determina la respuesta a la frecuencia de una red de potencia, inyectando una corriente de uno en por unidad o un voltaje en el nodo de interés con pasos de frecuencia para un rango en particular. El método de fuente de corriente más simple usa el marco de referencia de componentes de secuencia para obtener la propagación de las corrientes armónicas características, inyectando corrientes con fuentes ideales en la red

[Mahmoud y Schultz 1982]. En una contribución posterior, la solución de sistemas de potencia se obtiene directamente en el marco de referencia de fases para sistemas trifásicos [Howroyd 1982]. Un sistema que tenga excitación híbrida, es decir, que contenga voltajes armónicos en algunos nodos e inyecciones de corriente en otros; se resuelve dividiendo la matriz de admitancia y realizando una inversión parcial. Este procedimiento de solución híbrida permite que los voltajes nodales y corrientes armónicas desconocidas sean encontrados. En [Medina y Arrillaga 1990] se presenta una técnica eficiente para la solución de redes no lineales que contengan excitaciones de voltaje y corriente.

El análisis armónico iterativo está basado en sustituciones secuenciales del tipo Gauss. Los elementos que producen armónicos son modelados como una fuente de corriente dependiente de voltaje; representada por una fuente de corriente de armónicos fija en cada iteración. Las corrientes armónicas son obtenidas usando una fuente de voltaje estimada, entonces las corrientes armónicas son usadas para obtener los voltajes, estos voltajes en turno permiten el cálculo de corrientes armónicas más precisas. El proceso de solución se detiene una vez que los cambios en las corrientes armónicas son suficientemente pequeños [Callaghan y Arrillaga 1990].

Los métodos de flujos de potencia armónicos toman en cuenta la naturaleza depedendiente del voltaje de las componentes de potencia. En general, las ecuaciones de voltajes y corrientes armónicos se resuelven simultáneamente usando algoritmos del tipo Newton [Xia y Heydt 1982]. Los elementos no lineales y variantes en el tiempo son incorporados por medio de sus equivalentes Norton [Arrillaga y Medina 1995].

En [Collins *et. al* 2006] se presenta el modelo del UPFC en el dominio de la frecuencia [Arrillaga y Medina 1995], el modelado en el dominio de la frecuencia tiene ventajas que se presentan en [Arrillaga *et.* al 2000], por ejemplo: el proceso de linealización alrededor del punto de operación es relativamente simple, la representación explicita del acoplamiento armónico, fases y efectos del desbalance de fases y una solución iterativa unificada para redes lineales o no lineales. Una de las principales desventajas de estos métodos son los altos requerimientos de memoria asociados con la representación de los sistemas así como el gran esfuerzo computacional durante el proceso de solución, el cual involucra inversiones matriciales de matrices dispersas de gran tamaño, otra desventaja es que la precisión de la solución está ligada al número de armónicos que se desean considerar, lo que consecuentemente incremente los requerimientos computacionales.

Métodos híbridos (Frecuencia-Tiempo)

Son métodos que conjugan las ventajas de los métodos en el dominio del tiempo y el dominio de la frecuencia. En [Usaola 1990] se presenta un método de análisis híbrido en el

cual la parte lineal de la red se analiza en el dominio de la frecuencia en tanto que la parte no lineal se trata en el dominio del tiempo. En este trabajo se incorpora una modificación de la técnica de acercamiento rápido al EEP propuesta en [Aprille y Trick 1972] para el acercamiento rápido al EEP de la parte no lineal. En [Semlyen y Medina 1995] se propone una técnica de análisis en dónde la red es analizada por separado, por un lado los elementos lineales y los dependientes de la frecuencia se analizan en el dominio de la frecuencia, mientras que los elementos no lineales y variantes en el tiempo se analizan en el dominio del tiempo; en este trabajo se proponen tres métodos Newton para alcanzar la rápida convergencia hacia el EEP de la parte no lineal del sistema. Las técnicas propuestas tanto en [Usaola 1990] como en [Semlyen y Medina 1995] son aplicadas en el análisis de sistemas eléctricos monofásicos con componentes no lineales. La metodología propuesta por [Semlyen y Medina 1995] se aplica a una red trifásica simple en [Semlyen y Shlash 2000].

En [Medina y Arrillaga 1990] se presenta un método híbrido eficiente, que hace uso y aprovecha las ventajas de las técnicas de dispersidad y puede ser utilizado en cualquier problema que requiera la formulación de ecuaciones híbridas para describir la operación de un sistema.

1.1.2. Técnicas de procesamiento en paralelo

El principal factor del auge del procesamiento en Tesisparalelo, es la posibilidad de resolver problemas de gran escala descomponiendo el problema original en sub-problemas. El procesamiento en paralelo es definido en [Alvarado *et al*. 1992] como una forma de procesamiento de información en la cual dos o más procesadores juntos y con algún tipo de esquema de comunicación entre ellos, pueden cooperar para obtener la solución del problema. El procesamiento en paralelo se basa en dividir casi cualquier problema muy grande en problemas pequeños, los cuales pueden ser procesados en paralelo si los procesos involucrados son independientes entre si.

La solución alternativa a las limitaciones asociadas con las computadoras basadas en tecnologías de uni-procesadores, es el procesamiento en paralelo. Las limitaciones que presentan son: la frecuencia del reloj que no pueden superar la velocidad de la luz y la reducción del tamaño físico de los procesadores que está en función del tamaño de las moléculas [Jin 1994].

En [Ramos 2002] se hace un análisis detallado de dos plataformas operativas: Multithreading [Kleiman *et al*. 1992] y PVM [Geist *et al*. 1994]; la primera consiste en un esquema multiprocesadores que separa un proceso en muchos hilos de ejecución, cada uno se ejecuta por separado, pero comparten la memoria de la computadora. La segunda consiste en un esquema multicomputadoras que convierte una red heterogénea de computadoras en una gran computadora muliprocesadores.

Con la incorporación de técnicas de procesamiento en paralelo se tiene una reducción del tiempo de cómputo al incrementar el tamaño de los problemas, así como tener la posibilidad de realizar estudios en tiempo real y aumentar el detalle de los modelos que representan a los sistemas eléctricos de potencia.

En [Stavrakakis *et. al* 1990] se presenta un algoritmo en paralelo para la detección de fallas en manejadores de motores de corriente directa. En [Mariños *et. al* 1994] se hace aplicación de procesamiento en paralelo al análisis de armónicos en sistemas eléctricos de potencia.

En la contribución hecha por [Garcia *et. al* 2001] se presenta la aplicación de técnicas de procesamiento en paralelo a las técnicas Newton de acercamiento rápido al ciclo límite de las variables de estado. Posteriormente, en [Garcia y Acha 2004] se presentan las técnicas de acercamiento rápido al Ciclo Límite de las variables a la red de 118 nodos trifásico modificado.

La implementación de técnicas de procesamiento en paralelo no es nueva en el análisis de sistemas eléctricos de potencia, en el trabajo de [Medina y Ramos 2005b] se aplican las técnicas de procesamiento en paralelo basadas en dos plataformas distintas, al análisis de la respuesta a la frecuencia de redes eléctricas. En la contribución hecha en [Medina *et al*. 2006] se aplican técnicas de procesamiento en paralelo utilizando Multihreading a las técnicas Newton de acercamiento rápido al ciclo límite de las variables de estado de una red eléctrica no lineal.

1.1.3. Técnicas de dispersidad

La solución en redes eléctricas requiere de técnicas computacionales eficientes a fin de aprovechar de la mejor manera posible la memoria disponible, así como el reducir el tiempo de computo. Para este efecto hay que recordar que la estructura matricial que se obtiene de la formulación nodal de los sistemas eléctricos de potencia es muy dispersa, ya que no todos los elementos se encuentran conectados entre si. El manejo eficiente de la información de una matriz con esta propiedad, es decir, una matriz dispersa, se ve reflejado en la optimización de recursos tales como esfuerzo computacional y manejo de memoria, respectivamente.

En el trabajo de [Tinney y Walker 1967] se obtienen ganancias en velocidad, requerimientos de memoria y reducción de errores por redondeo, ya que el problema $Ax = B$ se soluciona mediante un proceso de factorización triangular.

En [Medina y Arrillaga 1990] se incorpora las técnicas de dispersidad a un método en híbrido, en la contribución hecha por [Medina y Ramos 2005a] se presenta la incorporación de técnicas de dispersidad a un método de acercamiento rápido a la solución en EEP, que permite una rápida convergencia al EEP, los mismos autores en [Medina y Ramos 2005b] aplican las técnicas de dispersidad en el proceso de la determinación de la respuesta a la frecuencia de redes eléctricas.

En [Timothy 2006] se presentan algoritmos para realizar operaciones matriciales de forma dispersa, con manejo de memoria dinámica, las técnicas de dispersidad han evolucionado en función a los avances en computación, tales como el manejo de memoria dinámica o el uso de estructuras dinámicas.

1.2. Objetivos

Determinar y analizar las características y potencialidad del método MED aplicado a la obtención de la solución periódica en estado estacionario periódico de sistemas eléctricos no lineales.

Desarrollar una herramienta de simulación que permita la determinación del estado estacionario periódico de sistemas eléctricos no lineales en el dominio del tiempo, mediante un método Newton de solución basado en el método MED, que además incorpore técnicas de dispersidad y procesamiento en paralelo, basado en la plataforma de OpenMP.

1.3. Justificación

En el análsis y diseño de los sistemas eléctricos de potencia se requieren de simulaciones que muestren su comportamiento en estado estacionario periódico. Sin embargo, debido al número de Ecuaciones Diferenciales Ordinarias involucradas en su representación matemática detallada, las simulaciones suelen demandar un gran esfuerzo computacional y en consecuencia largos lapsos de tiempo de cómputo, por lo que se hace necesario contar con una herramienta que brinde la posibilidad de encontrar de una forma más rápida dicha solución.

Dentro de los distintos métodos y marcos de referencia disponibles en la actualidad para la determinación de la solución periódica en estado estacionario de redes eléctricas no lineales, en esta tesis se ha optado por elegir el marco de referencia del tiempo, y en particular por la aplicación del método MED, con el propósito de profundizar en sus características numéricas, eficiencia y potencial aplicación en el análisis de redes eléctricas no lineales.

Ahora bien, el considerable aumento en la rapidez de cálculo de los distintos métodos de análisis está estrechamente relacionado con el aumento de la capacidad de procesamiento de las computadoras. De este modo, la efectividad final de un método depende en gran medida de la destreza con que el programador pueda manejar los algoritmos de resolución para la solución del estado estacionario periódico de sistemas eléctricos no lineales. Por lo anterior, se pretende desarrollar una herramienta digital que incorpore de manera óptima las ventajas que nos brindan, los algoritmos numéricos y la tecnología de procesamiento en paralelo para llegar a la solución en el menor tiempo posible.

1.4. Metodología

La investigación reportada en esta tesis, está basada en la realización del siguiente procedimiento metodológico:

1. Revisión del estado del arte sobre procesamiento en paralelo, técnicas de dispersidad y sobre el método Newton basado en la Matriz Exponencial Discreta.

2. Desarrollo de modelos monofásicos de la red de potencia en el dominio del tiempo, así como su implementación.

3. Programación del método Newton basado en la Matriz Exponencial Discreta, escrito en lenguaje C.

4. Incorporación de técnicas de procesamiento en paralelo.

5. Incorporación de técnicas de dispersidad, utilizando la biblioteca de CSparse.

6. Comparar los resultados obtenidos y los tiempos de cómputo.

7. Publicación de los resultados.

1.5. Descripción de los capítulos de la tesis

En el Capítulo 1 se presenta una introducción a esta tesis y se hace una reseña de los antecedentes en las técnicas de acercamiento rápido al estado estacionario periódico, así como algunos trabajos que han implementado técnicas de procesamiento en paralelo y técnicas de dispersidad al análisis de sistemas eléctricos. Se describe la justificación para el desarrollo del presente trabajo y se fijan los objetivos de esta tesis. Finalmente se describe el contenido de cada capítulo.

En el Capítulo 2 se presenta un análisis de dos técnicas de acercamiento rápido al estado estacionario periódico en el dominio del tiempo: Diferenciación Numérica y Matriz Exponencial Discreta. Se describen las diferentes plataformas y arquitecturas que tienen las técnicas de procesamiento en paralelo, haciendo en particular un análisis detallado de la plataforma de OpenMP. Se describen también las técnicas para el manejo de matrices dispersas.

En el Capítulo 3 se presenta una descripción del método Newton basado en la Matriz Exponencial Discreta. Se detalla la incorporación de las herramientas computacionales de técnicas de procesamiento en paralelo y técnicas de dispersidad durante el proceso de solución.

En el Capítulo 4 se presentan casos de estudio utilizando redes eléctricas monofásicas de mediana escala con el propósito de comparar los métodos de Diferenciación Numérica

y el de la Matriz Exponencial Discreta. Además se detallan los resultados obtenidos de la incorporación de herramientas computacionales avanzadas al método de MED.

En el Capítulo 5 se presentan conclusiones generales que son resultado del trabajo realizado en ésta investigación y se dan las sugerencias para trabajos de investigación futuros.

Capítulo 2

Técnicas: Numéricas y computacionales para la determinación del estado estacionario periódico

2.1. Introducción

La incorporación de poderosas herramientas de cómputo, matemáticas y numéricas, ha permitido reducir el tiempo de cómputo en las simulaciones de los sistemas eléctricos de potencia, compuestos por elementos lineales, no lineales y variantes en el tiempo, ante diferentes escenarios de operación. El incremento de cargas y componentes no lineales y variantes en el tiempo, tales como convertidores de potencia; así como la presencia de fenómenos no lineales, tales como la saturación magnética en los transformadores, máquinas rotatorias, etc., ha aumentado la distorsión armónica de las formas de onda de corriente y voltaje en los sistemas eléctricos de potencia.

La solución periódica en estado estacionario de un sistema de potencia se puede determinar mediante distintas metodologías desarrolladas en el dominio de la frecuencia, en el dominio del tiempo y en dominio híbrido frecuencia-tiempo [Kundert *et al*. 1990]. En [Kundert *et al*. 1990] se hace análisis de las características, ventajas y desventajas principales, asociadas con cada metodología.

En esta tesis se aplica el método de la Matriz Exponencial Discreta [Segundo y Medina 2010a] en el dominio del tiempo para la determinación del estado estacionario periódico de sistemas eléctricos no lineales. En las siguientes secciones del presente capítulo, se describen las herramientas numéricas y computacionales avanzadas utilizadas en el desarrollo de esta tesis. Estas son:

- Métodos Newton basados en mapa de Poincaré y extrapolación al Ciclo Límite

- Técnicas de dispersidad
- Técnicas de procesamiento en paralelo

2.2. Métodos Newton basados en mapa de Poincaré y extrapolación al Ciclo Límite

El comportamiento dinámico de un sistema eléctrico de potencia representado en variables instantáneas, está descrito por un modelo matemático que consiste en un conjunto no autónomo de ecuaciones diferenciales ordinarias no lineales. Así que la solución en EEP del sistema eléctrico puede obtenerse directamente integrando las ecuaciones diferenciales ordinarias (EDO's) que modelan al sistema durante un lapso de tiempo suficientemente grande como para que el transitorio pueda ser despreciable [Parker y Chua 1989].

Para sortear los inconvenientes e ineficiencia computacional de los métodos de FB y obtener la solución en EEP de sistemas eléctricos no lineales, se han propuesto otros métodos basados en el método Newton y el mapa de Poincaré [Parker y Chua 1989].

Estos métodos de solución en el dominio del tiempo, se pueden utilizar para obtener el punto inicial del Ciclo Límite, independientemente de su estabilidad [Aprille y Trick 1972] y poseen una poderosa característica de convergencia, generalmente cuadrática.

2.2.1. Método Newton para el cálculo del Estado Estacionario Periódico basado en el Mapa de Poincaré

Considere el siguiente sistema no lineal variante en el tiempo,

$$\dot{x}(t) = f(x,t); \qquad x_0 = x\,(t_0) \tag{2.1}$$

donde x y f son vectores de dimensión n, y $f(t,x)$ es contínua en un rango de t. Si se considera que el sistema es excitado periódicamente con una función de periodo T, entonces la solución en EEP de las variables de estado serán también periódicas y del mismo periodo de excitación T.

La formulación necesaria puede obtenerse si se considera una perturbación sobre la trayectoria original que sigue las variables de estado, tal que x^i es perturbada como $x^i + \Delta x^i$ o también se puede expresar como $x(t) + \Delta x(t)$ en vez de $x(t)$, entonces la Ecuación (2.1) se transforma en:

$$\dot{x} + \Delta\dot{x} = f(x + \Delta x, t) \tag{2.2}$$

De 2.2 se obtiene,

$$\Delta \dot{x} \approx J(t)\Delta x \tag{2.3}$$

donde $J(t)$ es la matriz Jacobiana y está dada por

$$J(t) = D_x f(x,t) \tag{2.4}$$

Una órbita transitoria se puede describir por su trayectoria a través de un hiperplano imaginario que corta perpendicularmente a los ciclos de dicha órbita. A este plano se le conoce como Plano de Poincaré [Parker y Chua 1989].

La Figura 2.1 muestra esquemáticamente la trayectoria asociada a un sistema contínuo y periódico proyectada en el Plano de Poincaré (ρ):

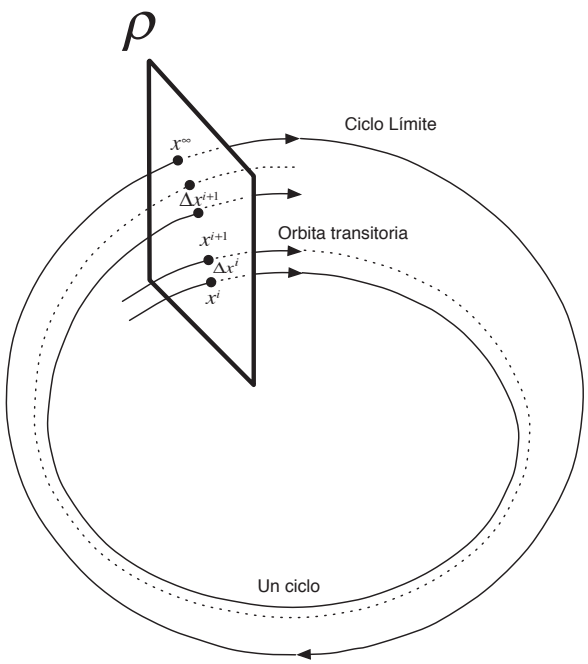


Figura 2.1: Plano de Poincaré [Semlyen y Medina 1995]

Sea $\varphi(t, t_0, x_0)$ la solución de (2.1) en el tiempo t con condición inicial $x_0 = x(t_0)$:

$$\varphi(t, t_0, x_0) = \int_{t_0}^{t} f(x,\tau)d\tau + x_0 \tag{2.5}$$

El Plano de Poincaré $\rho = \mathbb{R}^n \to \mathbb{R}^n$ se define por

$$\rho(x_0) = \varphi(t + T, t_0, x_0) \tag{2.6}$$

donde T es el periodo y es el tiempo que le toma de x^i a x^{i+1}, esto se puede observar en la Figura 2.1.

Los ciclos de la órbita transitoria que interceptan al plano cada periodo T son descritos matemáticamente por el Mapa de Poincaré [Parker y Chua 1989], el cual transforma un sistema periódico en un sistema discreto de punto fijo.

El mapa de Poincaré traslada el vector de variables de estado en el tiempo t al tiempo $t + T$, y de ésta manera queda definido el siguiente sistema discreto equivalente al sistema (2.1):

$$x^{i+1} = \rho(x^i) \tag{2.7}$$

donde x^i representa el vector de variables de estado en el tiempo discreto i.

La expresión (2.7) implica que la solución en EEP corresponde a un punto de equilibrio en el mapa de Poincaré, lo cual transforma la búsqueda de la solución en EEP a la búsqueda de un punto de equilibrio en el mapa de Poincaré, es decir, encontrar el vector x^i tal que $g(x^i) = 0$, implica resolver (2.7).

$$x^{i+1} - \rho(x^i) = g(x^i) \tag{2.8}$$

La solución de la Ecuación (2.8) puede resolverse iterativamente usando un método Newton como:

$$x^{\infty} = x^i + (I - \Phi)^{-1} \left(x^{i+1} - x^i\right) \tag{2.9}$$

donde,

x^{∞}	Variables de estado en el Ciclo Límite.
x^i	Variables de estado al inicio del ciclo base.
x^{i+1}	Variables de estado al final del ciclo base.
$\Phi = \frac{\partial \rho(x)}{\partial x}$	Matriz de transición de estados: que permite conocer $\Delta x(t)$ en cualquier tiempo t.
I	Matriz identidad.

x^{i+1} puede ser obtenida a través de la integración de x^i en un periodo T. Los procedimientos propuestos para calcular Φ serán detallados en la siguiente sección. Una vez que x^{∞} ha sido calculada utilizando (2.9), x^i es igualada a x^{∞} y la iteración (2.9) es repetida hasta que dos vectores de estado consecutivos converjan según el criterio de convergencia establecido.

2.2.2. Métodos para el cálculo de Φ

La parte más importante del método Newton, quizás sea el cálculo del Jacobiano del mapa de Poincaré Φ. La estabilidad del método, así como la eficiencia en el cálculo de la solución en EEP dependerá mucho del proceso bajo el cual se obtenga . En [Aprille y Trick 1972, Nayfeh 1995, Semlyen y Medina 1995, Banerjee y Verghese 2002] se describen varios métodos que permiten el cálculo de Φ. En [Aprille y Trick 1972] se presenta un método que permite el cálculo de Φ con la integración de (2.1) durante un periodo.

En [Semlyen y Medina 1995] se presenta un método basado en Diferenciación Numérica (DN), el cual requiere de la integración de $n+1$ periodos, donde n es la dimensión del conjunto de EDO's.

En los trabajos de [Semlyen y Medina 1995, Banerjee y Verghese 2002] se presenta un método basado en el análisis de sensibilidades, al que se le ha llamado método de Aproximación Directa (AD) [Semlyen y Medina 1995], éste método requiere integrar el mismo número de periodos que DN para el cálculo de Φ. Más recientemente en [Segundo y Medina 2010a] se reporta el método Matriz Exponencial Discreta (MED); que solo requiere de un periodo de integración en cada identificación de Φ.

La Figura 2.2 , muestra un diagrama de la metodología general para obtener el ciclo límite aplicando técnicas Newton.

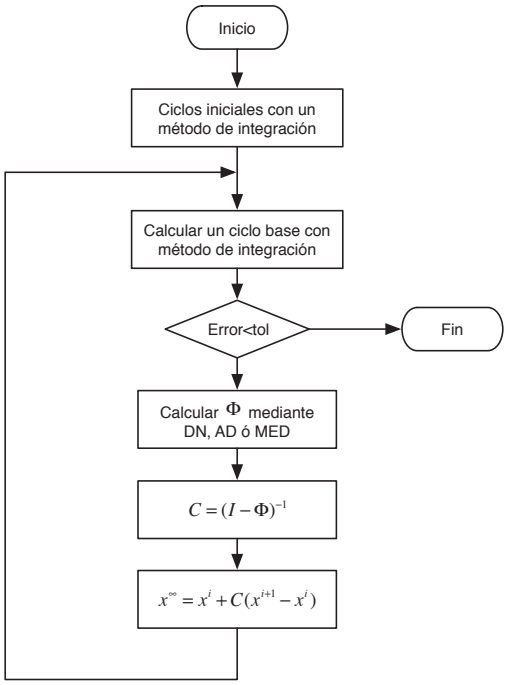


Figura 2.2: Diagrama de la solución en EEP aplicando métodos Newton

2.2.2.1. Diferenciación Númerica

Se debe de integrar durante un periodo T (2.1) con condiciones iniciales x_0 para obtener el Ciclo Base $x^i(t)$ [Semlyen y Medina 1995]. El vector de conidciones iniciales $x_0^{'}$, se perturba por una pequeña cantidad ε, obteniéndose $x_0^{'} + \varepsilon e_i$, donde e_i es la columna i de la matriz identidad I , y ε es una constante con valor pequeño, por ejemplo de 1×10^{-6}[Semlyen y Medina 1995].

Integrando (2.1) por un periodo T y con condiciones inciales $x_0^{'} + \varepsilon e_i$ se obtiene $x^{i'}$ y se calcula la diferencia obteniendo las columnas de la matriz ΔX^{i+1} como:

$$\Delta X^{i+1} = x_i^{'} - x^i \tag{2.10}$$

Con esta vector se obtiene Φ_i que es la columna i de la matriz de transición,

$$\Phi_i = \frac{\Delta X^{i+1}}{\varepsilon} \tag{2.11}$$

La evaluación de (2.11) se debe hacer para calcular cada una de las columnas de Φ, este procedimiento se resume mediante el diagrama de flujo mostrado en la Figura 2.3, el cual describe el método de Diferenciación Numérica, en donde se muestra la forma en la que se realiza el cálculo de las columnas de la matriz Φ.

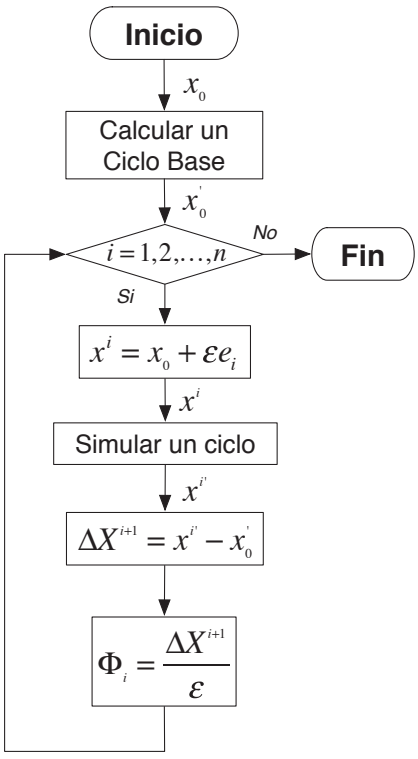


Figura 2.3: Diagrama del método de Diferenciación Numérica

2.2.2.2. Matriz Exponencial Discreta

En [Armanazi 1973] el cálculo de Φ se logra a través de productos de matrices exponenciales, limitado a sistemas lineales conmutados. En [Segundo y Medina 2010a] se propone el método de Matriz Exponencial Discreta (MED), en el que se muestra que Φ puede ser aproximado mediante productos exponenciales de matriz, incluso para sistemas conmutados no lineales.

La dinámica de (2.1) en las cercanías del ciclo límite puede ser aproximada por medio de,

$$\Delta\dot{x}(t) = J(t)\Delta\mathrm{x} \tag{2.12}$$

donde Δx es la desviación del vector de variables de estado respecto al Ciclo Límite y $J(t)$ es el Jacobiano de $f(t, x)$.

La solución de (2.12) expresada en tiempos discretos, múltiplos de T es,

$$\Delta x^{k+1} = \Phi^k \Delta x^k \tag{2.13}$$

donde,

$$\Delta x^k = \Delta x(kT) \qquad k = 0, 1, 3, ..., m \tag{2.14}$$

y

$$\Phi^k = \exp^{\int_{kT}^{(k+1)T} J(t)dt} \tag{2.15}$$

La Ecuación (2.15) es solución de (2.12) si $J(t)$ es de dimensión uno. Φ^k es la matriz de transición discreta de (2.12). Para nuestro propósito es posible considerar que alrededor del ciclo límite es aplicable la siguiente aproximación de Φ^k,

$$\Phi^k \approx \Phi^{k+1} \approx \Phi^{k+2} \approx \Phi^{k+3} \ldots \approx \Phi^{k+m} \approx \Phi \tag{2.16}$$

Sin embargo, si el Jacobiano es de dimensión superior a uno, entonces (2.15) no es solución de (2.12), a menos que el Jacobiano sea constante. Para propósitos prácticos, en el caso más general donde el Jacobiano $J(t)$ sea de cualquier dimensión y variante en el tiempo, es posible considerar a $J(t)$ constante durante pequeños intervalos de tiempo, de tal manera que la solución de (2.12) para el caso multivariable puede ser aproximada como [D'Angelo 1970],

$$\Delta x^{k+1} = e^{J_N \Delta \mathrm{t}_N} e^{J_{N-1} \Delta \mathrm{t}_{N-1}} \ldots e^{J_2 \Delta \mathrm{t}_2} e^{J_1 \Delta \mathrm{t}_1} \Delta x^k \tag{2.17}$$

donde Δt_i se define como $t_i + t_{i-1}$, N es el número de intervalos en un periodo y J_i es,

$$J_i = f_x\left(\left(t_i + t_{i-1}\right)/2,\ \left(x_i + x_{i-1}\right)/2\right) \tag{2.18}$$

t_i representa el i-ésimo elemento del vector tiempo desde t hasta $t+T$, también x_i es el vector de estado correspondiente al tiempo t_i.

La Ecuación (**??**) puede ser aproximada directamente como,

$$\Phi \approx \frac{\Delta \mathrm{x}(t+T)}{\Delta \mathrm{x}(t)} \tag{2.19}$$

Comparando (**??**), (2.17) y (2.19) está claro que tiene Φ la forma,

$$\Phi = \prod_{i=1}^{N} e^{J_i \Delta \mathrm{t}_i} \tag{2.20}$$

La expresión (2.20) representa la matriz de transición aproximada Φ basada en una expansión exponencial discreta [D'Angelo 1970]. Esta se aplicará para obtener la solución periódica en estado estacionario de sistemas no lineales a través del método de Newton basado en el mapa de Poincaré y la extrapolación al procedimiento del ciclo límite. Adicionalmente, esta formulación no está sujeta al método de integración numérico utilizado, como en el caso de [Aprille y Trick 1972], en cambio, (2.20) permite cualquier tipo de método de integración para ser utilizado para la determinación de Φ. Se puede notar que si $J(t)$ es un segmento invariante en el tiempo, entonces la matriz de transición discreta exacta de (2.12) puede ser calculada utilizando (2.20) [D'Angelo 1970]. La ecuación (2.20) implica que el sistema no lineal representado por (2.1) es aproximado por sistemas lineales sucesivos en cada paso de integración.

2.3. Técnicas de procesamiento en paralelo

El procesamiento en paralelo consiste en tener una gran cantidad de elementos de proceso trabajando en forma simultánea. Puede ser definido como una forma de procesamiento de información en la cual dos o más procesadores juntos o con algún tipo de esquema de comunicación entre ellos, pueden cooperar para obtener la solución de un problema [Alvarado *et al*. 1992]. El rápido desarrollo que se ha dado en el procesamiento en paralelo ha sido motivado por dos factores: la creciente necesidad de resolver problemas cada vez más complicados y los impresionantes avances en la tecnología computacional.

La mayoría de las computadoras de propósito general están diseñadas en base al uso de un sólo procesador. Esta tecnología tiene una serie de limitaciones físicas, tecnológicas y económicas, por lo que las computadoras tradicionales uni-procesadores son incapaces de satisfacer

las necesidades computacionales para aplicaciones cada vez más complejas. Una característica importante de la computación paralela es la estrecha relación entre los algoritmos y las arquitecturas computacionales paralelas.

A continuación se describen los principales conceptos de las técnicas de procesamiento en paralelo.

2.3.1. Diseño de algoritmos en paralelo

La mayoría de los problemas de programación tienen distintas soluciones paralelas. A continuación se describE una metodología para el diseño de algoritmos paralelos propuesta en [Foster 1994]. Esta metodología está basada en cuatro procesos: Partición, Comunicación, Agrupación, Asignación, los cuales se ilustran en la Figura 2.4.

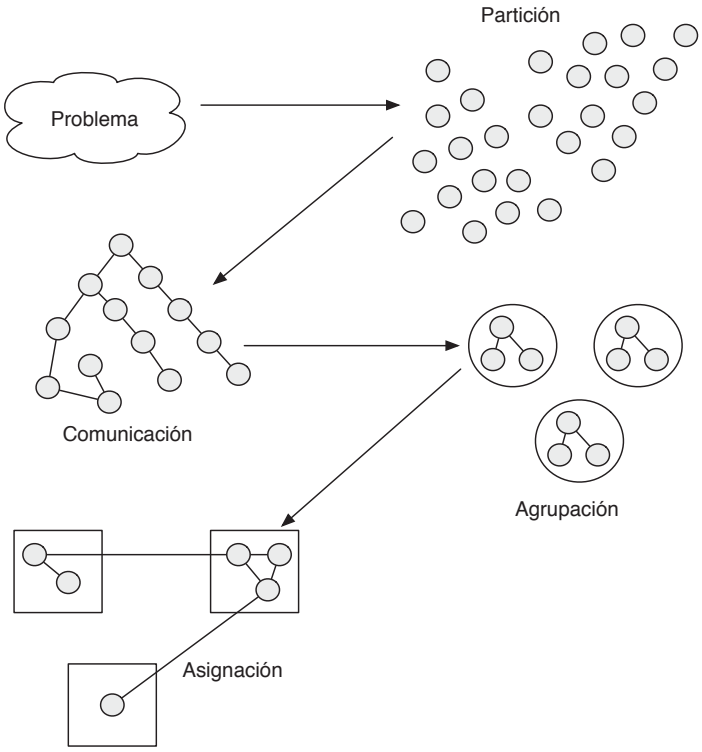


Figura 2.4: Estrategia de Paralelización de Algoritmos

A continuación se describen los cuatros procesos ilustrados en la Figura 2.4:

1. *Partición*: El cálculo que será realizado y los datos utilizados para este cálculo se descomponen en pequeñas tareas.

2. *Comunicación*: Aquí se establece la comunicación requerida para coordinar la ejecución de las tareas, además se definen los algoritmos y estructuras de comunicación apropiadas.

3. *Agrupación*: Las tareas y estructuras de comunicación definidas en las dos etapas anteriores se evalúan con respecto a los requerimientos de funcionamiento y costos de implementación. Si es necesario, las tareas se combinan en tareas grandes para mejorar el funcionamiento o reducir los costos de desarrollo.

4. *Asignación*: Cada tarea se asigna a un procesador tratando de maximizar la utilización de los procesadores y minimizar los costos de comunicación. El mapeo puede ser especificado en forma estática o determinarse en el tiempo de ejecución por medio de algoritmos de carga balanceada.

2.3.2. Medición del desempeño de programas con procesamiento en paralelo

El tiempo de ejecución no es siempre la medida más conveniente por medio del cual se puede evaluar el funcionamiento y eficiencia de un algoritmo paralelo o secuencial. Tal como los tiempos de ejecución tienden a variar con el tamaño del problema, los tiempos de ejecución deben ser normalizados cuando se compara el funcionamiento de un algoritmo en problemas de diferentes tamaños.

2.3.2.1. Eficiencia relativa (Speed-Up)

La manera en la que se cuantifica la eficiencia de un algoritmo paralelo o secuencial es calculando la efectividad con la que utiliza los recursos computacionales de la computadora paralela. La eficiencia relativa se define como [Foster 1994],

$$E_{relativa} = \frac{T_1}{T_p} \tag{2.21}$$

Donde:

T_1 Tiempo de ejecución obtenido con un sólo elemento de proceso.
T_p Tiempo de ejecución obtenido con p elementos de proceso.

De la Ecuación (2.21) resulta claro que al reducir el tiempo de procesamiento en paralelo se incrementa la Eficiencia Relativa, pero en realidad es que a medida que se incrementa el número de procesadores en paralelo, también el tiempo de comunicación entre ellos será

mayor y la ganancia de términos de la Eficiencia Relativa comenzará a drecrecer como lo predice la ley de Amdahl, la eficiencia relativa ideal es $E_{relativa} = p$, donde p es el número de elementos de proceso.

2.3.2.2. Eficiencia

Una medida alternativa para el desempeño de un programa paralelo es la eficiencia. La eficiencia es una medida en términos de la fracción de tiempo de los procesadores consumen haciendo trabajo útil, es una relación métrica que puede algunas veces proveer de una cantidad más conveniente para medir el desempeño de un algoritmo.

La eficiencia se define como [Foster 1994],

$$E = \frac{E_{relativa}}{p} \tag{2.22}$$

El valor de la eficiencia muestra el grado en el que se está utilizando la capacidad de proceamiento en paralelo, una eficiencia ideal corresponde a $E = 1$.

2.3.2.3. Ley de Amdahl

El número de procesadores o la dependencia de información del algoritmo a implementarse puede limitar el grado de paralelismo, por lo tanto el tiempo de ejecución paralelo no se puede reducir indefinidamente empleando procesadores.

La ley de Amdahl establece que hay una sección que debe ser ejecutada en forma secuencial, y otra sección paralelizable, entonces el tiempo de ejecución del programa tendrá una aproximación asintótica hacia el tiempo de ejecución secuencial, esto quiere decir que independientemente del número de elementos de proceso que sean utilizados el tiempo de computo no decrecerá más [Lewis y Berg 1998]. El comportamiento del tiempo de ejecución utilizando procesamiento en paralelo se puede observar en la Figura 2.5.

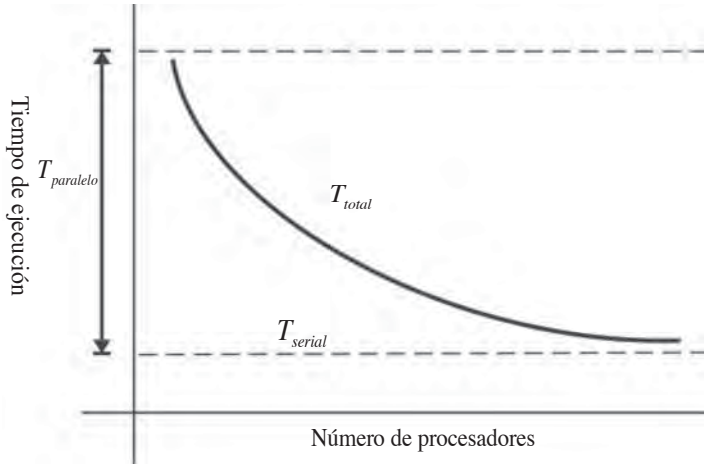


Figura 2.5: Tiempo de ejecución con múltiplos elementos de proceso

La Figura 2.5 muestra que el tiempo de ejecución en paralelo $T_{paralelo}$ no se puede reducir indefinidamente ya que existe una limitante que es el tiempo de ejecución secuencial $T_{secuencial}$. Este tiempo surge de la dependencia de información del algoritmo a implementarse lo cual puede limitar el grado de paralelismo. Otra restricción que puede surgir es el número de procesadores disponibles.

2.3.3. Enfoques del procesamiento en paralelo

En general, una computadora paralela puede caracterizarse como una colección de elementos de proceso que se pueden comunicar y cooperar para resolver rápidamente grandes problemas [Almasi y Gotlieb 1994]; sin embargo, ésta definición es muy amplia. Para realizar una investigación más detallada se puede identificar dos grandes enfoques en el procesamiento en paralelo:

- Paralelismo implícito o de bajo nivel
- Paralelismo explícito o de alto nivel

2.3.3.1. Paralelismo implícito o de bajo nivel

Las técnicas de paralelismo implícito son aquellas donde queda oculta la arquitectura de la computadora; por lo tanto, pueden ser vistos como sistemas de un sólo procesador. Se programa en lenguaje secuencial y el compilador se encarga de paralelizar y asignar los recursos correspondientes. La ventaja es que se aprovecha todo el código existente. La desventaja es la dependencia del compilador y el bajo nivel de aprovechamiento del paralelismo de la máquina [Quinn 2004].

2.3.3.2. Paralelismo explícito o de alto nivel

El paralelismo explícito es aquel donde se interconectan varios procesadores para cooperar en la ejecución de los programas de aplicación, por lo que el programador debe encargarse del uso adecuado de los operadores de control y es el responsable de realizar una paralelización de forma óptima.

La ventaja es que se da un mejor nivel de aprovechamiento del paralelismo de la máquina y ofrece una infraestructura explícita. La desventaja es que el trabajo del programador es más difícil, debido a los nuevos paradigmas de la programación [Quinn 2004].

2.3.4. Arquitecturas paralalelas

Cualquier arquitectura, ya sea secuencial o paralela, opera en base a ejecución de instrucciones sobre datos. El flujo de instrucciones (algoritmo) indica que es lo que se tiene que hacer y un flujo de datos (entrada al algoritmo) es modificado por éstas instrucciones.

Para una investigación más detallada, es útil hacer una clasificación de acuerdo a características importantes de una computadora paralela. Un modelo simple para dicha clasificación está dado por Flynn [Flynn 1972], el cual establece cuatro clases de arquitecturas:

Una sola instrucción, un solo dato o *Single Instruction Single Data* (SISD):

Una arquitectura de ésta clase sólo tiene un procesador, que recibe e interpreta un flujo de instrucciones simple en cada paso de la ejecución; la unidad de control proporciona una instrucción que opera con un dato obtenido de la unidad de memoria [Quinn 2004]. La Figura (2.6) muestra un esquema de la arquitectura SISD.

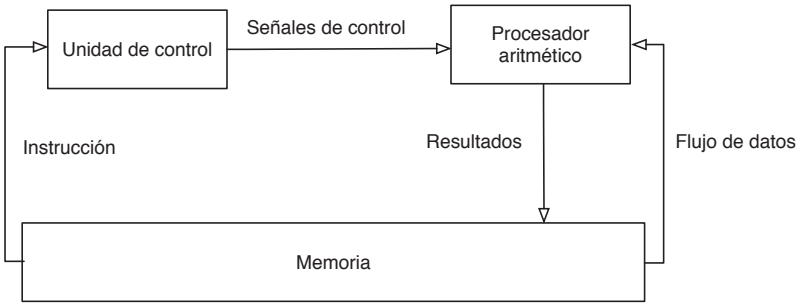


Figura 2.6: Arquitectura SISD

Una sola instrucción, múltiples datos o *Single Instruction Multiple Data* (SIMD):

Esta arquitectura consiste en tener n elementos de proceso, supervisados por una única unidad de control. Una sola instrucción es aplicada sobre diferentes datos al mismo tiempo. En las máquinas de este tipo un cierto número de elementos procesadores son controlados y sincronizados mediante una unidad de control.

Cada elemento de proceso tiene una memoria asociada, de manera que cada instrucción es ejecutada simultáneamente por todos los elementos procesadores pero sobre un conjunto de datos diferentes [Quinn 2004]. Debido a su utilidad en el procesamiento de matrices, a este tipo de máquinas se les llama procesadores matriciales. La Figura 2.7 muestra un esquema de la arquitectura SIMD.

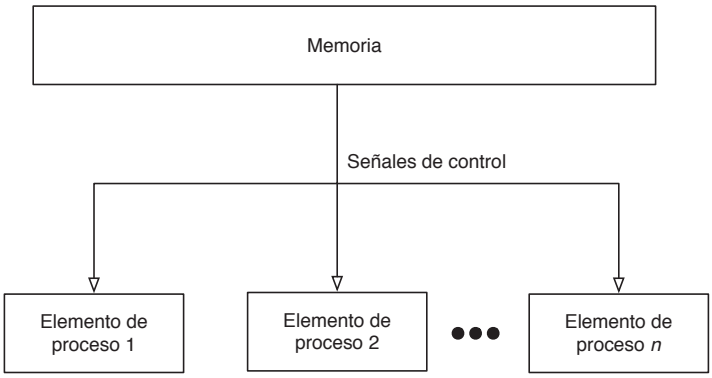


Figura 2.7: Arquitectura SIMD

Múltiples instrucciones, un solo dato o *Multiple Instruction Single Data* (MISD):

Esta arquitectura consta de n procesadores, cada uno cuenta con su propia unidad de control, cada una de ellas cuenta con su flujo de instrucciones y comparten una unidad de memoria común donde residen los datos. Entonces, hay n flujos de instrucciones y un sólo flujo de datos, por lo tanto, varias instrucciones actúan simultáneamente sobre un flujo único de datos. Se pueden considerar como máquinas formadas por varias unidades de procesamiento, las cuales reciben instrucciones distintas que operan sobre los mismos datos [Quinn 2004].

La Figura 2.8 muestra una representación de la arquitectura MISD.

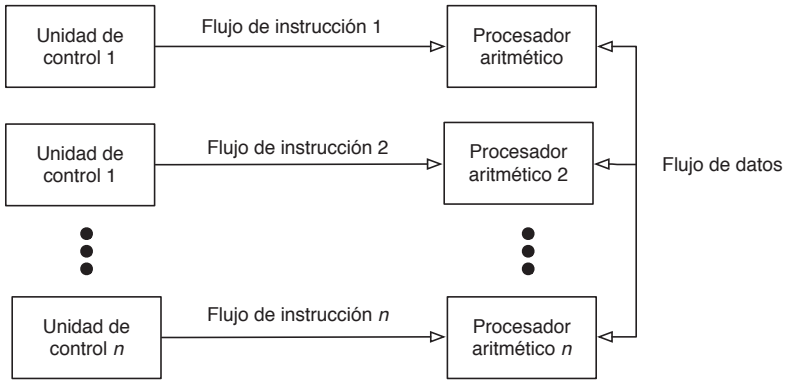


Figura 2.8: Arquitectura MISD

Múltiples instrucciones, múltiples datos o *Multiple Instruction Multiple Data* (MIMD):

Las computadoras MIMD son muy flexibles, ya que cada elemento de proceso puede ejecutar su propio flujo de programa. La mayoría de las computadoras paralelas están basadas en el concepto MIMD. Un conjunto de unidades de procesamiento ejecuta simultáneamente diferentes secuencias de instrucciones sobre conjuntos de datos diferentes. Se dispone de n procesadores, n flujos de instrucciones y n flujo de datos.

Cada procesador opera bajo el control de un flujo de instrucciones que le proporciona su propia unidad de control. Todos los procesadores ejecutan programas diferentes sobre diferentes datos mientras resuelven subproblemas de un único problema. Esto significa que los procesadores operan asíncronamente. De cara a la correcta resolución del problema, los diferentes procesos se tienen que comunicar y sincronizar. Esto se puede realizar mediante memoria compartida o por medio de una red de interconexión [Quinn 2004].

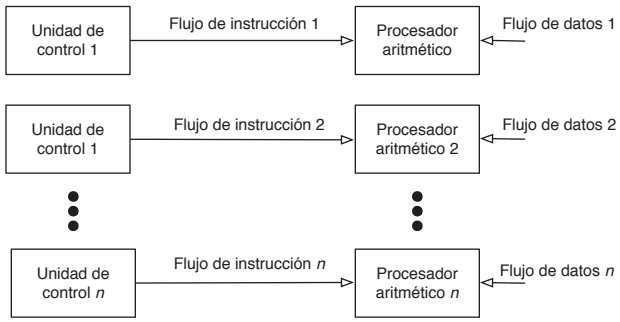
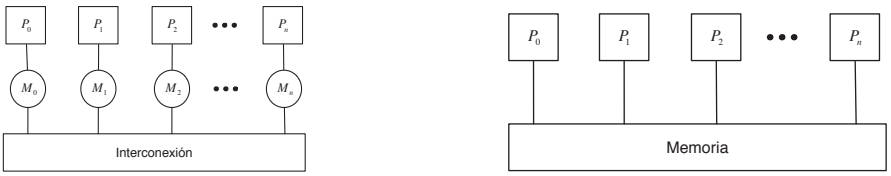


Figura 2.9: Arquitectura MIMD

Los computadores MIMD que comparten memoria común se les conoce como multiprocesadores o de memoria compartida (Figura 2.10b), mientras que los que utilizan una red de

interconexión se les conoce como multicomputadores o de memoria distribuida [Quinn 2004] (Figura 2.10a).



(a) Memoria Distribuida (b) Memoria Compartida

Figura 2.10: Diagramas de las arquitecturas basadas en MIMD

2.3.5. Plataformas operativas para el procesamiento en paralelo

Para mejorar el rendimiento de las aplicaciones sobre arquitecturas paralelas es necesario determinar el modelo de programación más adecuado.

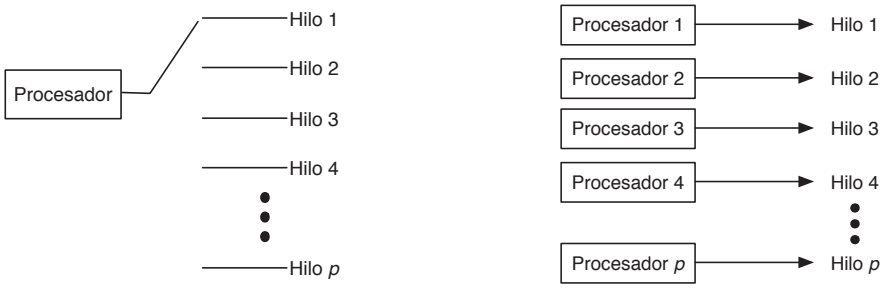
2.3.5.1. Multithreading

A un semi-proceso independiente que ejecuta un conjunto de instrucciones se le conoce como *thread* o hilo. El concepto de un "procedimiento" el cual se ejecuta independientemente dentro de un proceso, puede describir de mejor manera el concepto de hilo.

Actualmente un gran porcentaje de programas escritos en C/C++ aún utilizan un solo flujo de instrucciones (*thread*) para realizar su procesamiento. La palabra Multithreading se refiere a múltiples hilos de control o múltiples flujos de control y su ideología se basa en separar un proceso en muchos hilos de ejecución, cada uno de los cuales se ejecuta por separado.

Un proceso tradicional UNIX siempre contiene un solo hilo de control, Multithreading separa un proceso en muchos hilos de ejecución, cada uno de los cuales se ejecuta por separado [Kleiman *et al*. 1992]. En el ambiente de UNIX, existe un hilo dentro de un proceso el cual utiliza los recursos del proceso. Sin embargo, un hilo tiene su flujo de control propio e independiente y pueden existir múltiples hilos dentro de un proceso.

Un sólo procesador puede conmutar los recursos de ejecución entre los hilos, resultando en que la velocidad de conmutación entre los hilos es tan rápida que provoca que aparentemente todos los hilos se estén ejecutando de forma simultánea. En la Figura 2.11a muestra la forma en la que un sólo procesador conmuta para ejecutar p hilos.



(a) Ejecución concurrente de p hilos

(b) Ejecución en paralelo de p hilos

Figura 2.11: Ejecución de un proceso MT con un procesador y p procesadores

2.3.5.2. PVM

PVM (Máquina Paralela Virtual) es una biblioteca que permite a una colección heterogénea de computadoras conectada en red y funciona como si fuera una gran computadora [Geist *et al*. 1994].

Cuando PVM está instalado correctamente es capaz de combinar los recursos de varias computadoras, típicamente heterogéneas conectadas en red para entregar altos niveles de desempeño y funcionalidad. La Figura 2.12 muestra la representación conceptual de PVM.

El software PVM ha sido distribuido de forma gratuita y es utilizado en aplicaciones computacionales alrededor del mundo. El procesamiento en paralelo, método por el cual se resuelve un gran problema mediante muchas tareas pequeñas, ha emergido como una tecnología aplicada fundamental en la computación moderna.

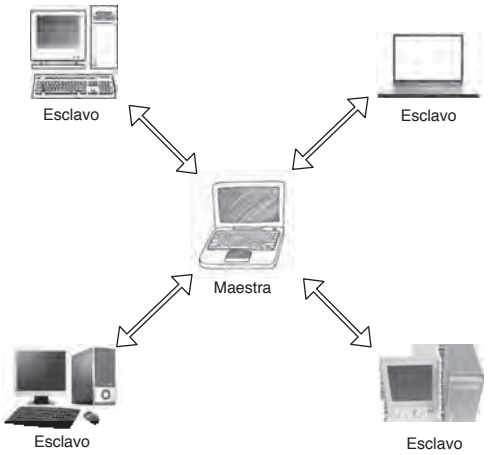


Figura 2.12: Representación conceptual de la plataforma PVM

2.3.5.3. MPI

MPI ("*Message Passing Interface*", Interfaz de Paso de Mensajes) es un estándar que define la sintáxis y la semántica de las funciones contenidas en una biblioteca de paso de mensajes.

Este modelo de programación puede ser utilizado tanto en arquitecturas de memoria compartida como en arquitecturas de memoria distribuida. El modelo de paso de mensajes en una máquina de memoria compartida puede implementarse utilizando las bibliotecas que provee el estándar MPI. Sin embargo, el uso de estas bibliotecas introduce grandes latencias que perjudican el rendimiento de la aplicación. Por esta razón, la plataforma de utilización de este estándar, está dirigida a sistemas de memoria distribuida incluyendo máquinas paralelas, SMP (Multiprocesadores simétricos) *clusters*, estaciones de trabajo y redes heterogéneas [Quinn 2004].

En el modelo de programación de paso de mensajes, el paralelismo es explícito debido a que el programador es el responsable de identificar correctamente el paralelismo e implementarlo utilizando los constructores adecuados que provee MPI. Sin embargo, existen factores que afectan el rendimiento de las aplicaciones distribuidas con paso de mensajes.

En arquitecturas de memoria compartida, las implementaciones de MPI, no utilizan una red de comunicaciones entre tareas, en su lugar, y por razones de rendimiento utilizan copias de memoria.

2.3.5.4. OpenMP

OpenMP es una colección de directivas del compilador y funciones de biblioteca que son utilizadas para crear programas paralelos en computadoras de memoria compartida, de hecho se ha vuelto el estándar de paralelización en este tipo de arquitectura. OpenMP se combina con el lenguaje C, C++ o Fortran para crear un lenguaje de programación multihilos, es decir, un lenguaje modelo que se basa en considerar que las unidades de ejecución o procesadores son hilos que comparten un espacio de dirección [Chandra *et al* 2001].

OpenMP está basado en el modelo de programación *Fork/Join* que se muestra en la Figura 2.13.

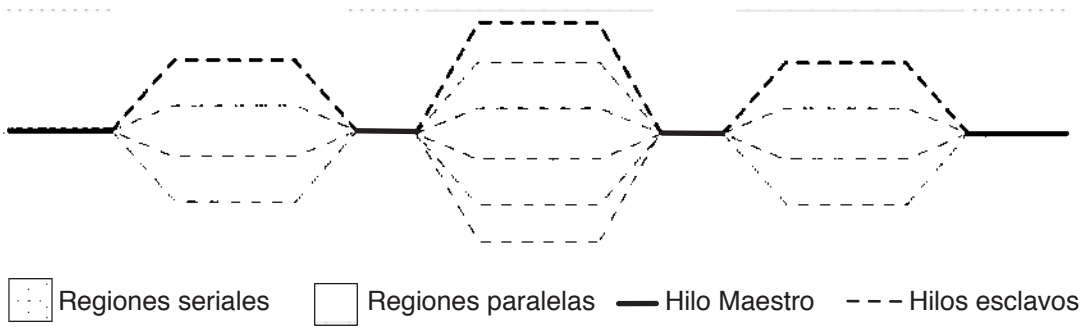


Figura 2.13: Representación del modelo *Fork/Join*

Una ejecución de un programa en OpenMP inicia como un simple hilo. En el punto del programa donde la ejecución en paralelo es deseable, el programa se ramifica en hilos adicionales para formar un conjunto de hilos esclavos. Los hilos se ejecutan en paralelo a través de una parte de código llamada *región paralela*. Al final de la región paralela, los hilos esperan hasta que todo el conjunto de hilos terminen su tarea asignada. En este punto, el hilo principal o master continúa hasta la próxima región paralela (o el final del programa).

OpenMP fue diseñado alrededor de dos conceptos clave:

- *Equivalencia secuencial*: Es cuando el programa rinde los mismos resultados si se ejecuta utilizando uno o más hilos.
- *Paralelismo incremental*: Es un estilo de programación paralela en la cual un programa evoluciona de un programa secuencial a un programa paralelo.

OpenMP comprende tres componentes básicos:

- Un conjunto de directivas de compilador usado por el programador para comunicarse con el compilador en paralelismo.
- Una biblioteca de funciones en tiempo de ejecución que habilita la colocación e interroga sobre los parámetros paralelos que se van a usar, tal como el número de los hilos que van a participar y el número de cada hilo.
- Un número limitado de las variables de entorno que pueden ser usadas para definir en tiempo de ejecución parámetros del sistema en paralelo, tales como el número de hilos.

Programar en OpenMP no es muy diferente a utilizar cualquier otra biblioteca de las existentes debido a que no es un nuevo lenguaje de programación; la diferencia es que cuando se está desarrollando una aplicación utilizando procesamiento en paralelo, no solamente tiene que establecer las entradas y salidas, sino que además se tiene que establecer la forma que será distribuido el trabajo entre los múltiples procesadores.

Conceptos Básicos de OpenMP

Cualquier paralelismo expresado en un programa está ahí porque el programador dirigió al compilador a "ponerlo ahí". Para crear hilos en OpenMP, el programador designa bloques de código para que sean ejecutados en paralelo.

OpenMP entra en un programa nuevo o existente a través de directivas, con funciones propias y variables de entorno, que pueden modificar el comportamiento en tiempo de ejecución. La mayoría de los bloques de construcción en OpenMP son directivas de compilación o pragmas; esto se hace en C y C++ con el pragma (*Pragmatic Information*) y tiene la forma:

```
#pragma omp directive-name[clause [clause]...]
```

donde la directive-name identifica la construcción y la cláusula opcional la modifica.

Al ejecutarse un programa en OpenMP se crea un número de hilos, si el programador no especifica el número de hilos a crear, se utiliza un número por *default,* y no hay un estándar en muchos sistemas; es decir,

$$\#\text{ de hilos} = \#\text{ de procesadores}$$

OpenMP es un modelo que contempla la siguiente regla: si se declara una variable dentro de la región paralela, debe especificarse si es local o privada para un hilo.

Construcción de regiones paralelas

Las siguientes directivas definen una región paralela, la cual es una parte del programa que puede ser ejecutada por múltiples hilos en paralelo. Éste es el constructor fundamental que inicia la ejecución paralela.

```
#pragma omp parallel [clause[,] clause]... { bloque de código }
```

Las cláusulas que se pueden utilizar se presentan en la Tabla 2.1:

Tabla 2.1: Cláusulas del constructor paralelo

Nombre:	Tipo:
if	Expresión escalar
private	Lista de variables
firstprivate	Lista de variables
default	Shared | none
shared	Lista de variables
reduction	Operador : lista de variables
num_threads	Expresión entera

De la Tabla 2.1 se pueden identificar las cláusulas que son más utilizadas:

- *private*: En ésta, la lista de variables puede ser separada por *private*, y para cada una de ellas se genera una copia en cada hilo; ésta copia no tiene relación con la original y no es inicializada a menos que se utilice la instrucción firstprivate.
- *shared*: En está, las variables de la lista son comunes a todos los hilos y cada uno de ellos puede modificarla afectándola en forma global.
- *threadprivate*: Hace que la lista sea privada a cada hilo pero globales dentro de un hilo.
- *reduction*: Realiza una operación de reducción sobre las variables que aparecen en la lista, utilizando el operador especificado.

Al final de una región paralela hay una sincronización implícita. Sólo el hilo maestro continua la ejecución. La Figura 2.14 muestra el proceso de cómo se construye una región paralela.

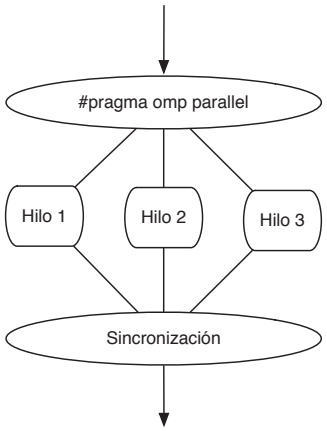


Figura 2.14: Construcción de regiones paralelas

Para determinar el número de hilos necesarios e identificarlos, existe la biblioteca Runtime; dentro de esta biblioteca, las funciones que se usan comúnmente son:

- *omp_set_num_threads*: Toma un argumento entero y requiere que el sistema operativo proporcione el número de hilos en las regiones paralelas subsecuentes.
- *omp_get_num_threads*: Función entera que regresa el número actual de hilos en el conjunto actual de hilos.

- *omp_get_thread_num*: Función entera que regresa el ID (identificador) de un hilo, donde el rango de ID va desde 0 hasta el número de hilos -1. El hilo con el ID de 0, es el hilo maestro.

Trabajo compartido.

Se le llama trabajo compartido cuando cada hilo ejecuta el mismo bloque de instrucciones, en OpenMP el tiempo compartido es comúnmente utilizado para dividir ciclos que el programador identifica que consumen más tiempo en el programa.

Los constructores que define OpenMP para trabajo compartido son [Quinn 2004]:

- Directiva *for*: Identifica un constructor de trabajo compartido, el cual especifica que las iteraciones del *loop* asociado deben ser ejecutadas en paralelo.
- Directiva *sections*: En la directiva sections cada sección es ejecutada una vez por un hilo en el grupo.
- Directiva *single*: Esta directiva se utiliza para referirse a que cada tramo del código va a ejecutarse por un sólo hilo.

Sincronización.

La sincronización es el proceso de administración de recursos compartidos de tal forma que cuando ocurra una lectura o escritura se haga en el orden correcto sin tomar en cuenta cómo los hilos están programados [Quinn 2004]. Las directivas principales son las siguientes :

- Directiva *master*: Especifica que el bloque de código es ejecutado por el hilo maestro, los demás saltan el bloque y continúan la ejecución.
- Directiva *critical*: Identifica que el bloque de código es accedido por un sólo hilo a la vez. Cada hilo espera el comienzo de bloque hasta que ningún otro hilo lo esté ejecutando.
- Directiva *barrier*: Cada hilo espera hasta que cada uno de los demás hilos termine.
- Directiva *atomic*: Es un caso específico del constructor critical que puede ser usada para ciertas declaraciones simples Ésta se utiliza sólo para la actualización de una localización en memoria.
- Directiva *flush*: Define un punto de sincronización en el cual la consistencia de la memoria es forzada. Esto puede ser delicado. Básicamente, una computadora moderna

puede soportar valores en registros o buffers que no garantizan ser consistentes con la memoria de la computadora en cualquier punto dado. Los protocolos de la coherencia cache garantizan que todos los procesadores finalmente vean un simple espacio de dirección, pero éstos no garantizan que las referencias a la memoria sean actualizadas y consistentes en cada punto a tiempo.

Cláusula de calendarización.

La clave para ejecutar un ciclo basado en un algoritmo en paralelo es programar (calendarizar) las iteraciones del ciclo sobre los hilos, de tal forma que la carga esté balanceada entre los hilos. Esto se cumple añadiendo la cláusula schedule a la construcción de trabajo compartido for. Las categorías utilizadas en esta cláusula puede ser una de las siguientes:

- *static*: cuando ésta es usada, el número de iteraciones es dividido en tramos y es asignado en forma de Round Robin.
- *dynamic*: cuando se usa esta categoría, la cantidad de iteraciones es dividida en una serie de tramos. Cada hilo ejecuta su tramo y espera por más trabajo asignado.
- *guided*: para esta categoría las iteraciones son asignadas a los hilos en tramos con tamaño decreciente. Cuando un hilo finaliza el tramo de iteraciones asignado, al hilo se le asigna otro tramo dinámicamente hasta que se termine la repartición.
- *runtime*: cuando se usa *runtime*, las categorías de schedule son obtenidas en tiempo de ejecución con la configuración dada en la variable de entorno OMP_SCHEDULE.

2.4. Técnicas de dispersidad

La solución de sistemas eléctricos de potencia con elementos lineales y no lineales, requiere de técnicas computacionales eficientes a fin de aprovechar de la mejor manera posible la memoria disponible, así como el reducir el tiempo de cálculo. Para este efecto se reconoció, a principios de los años 60 [Sato y Tinney 1963], que la formulación nodal lleva una estructura matricial con muchos ceros ya que no todos los nodos se encuentran conectados entre sí.

Una matriz dispersa es aquella cuyos elementos son en su mayoría ceros. Existen varias maneras de almacenar una matriz dispersa. Con cualquier método que se escoja, se requiere alguna forma de estructura de datos compacta que evite guardar los valores que son cero en la matriz de coeficientes. Se necesita que sea simple y flexible de esa manera puede utilizarse en varios tipos de operaciones matriciales. Esta necesidad la cumple la estructura de datos

primaria en CSparse, una matriz de columna condensada. En las siguientes sub-secciones se describen las operaciones matriciales básicas que trabajen en esta estructura de datos.

2.4.1. Estructura de datos de matrices dispersas

La estructura de datos de matrices dispersas más sencilla es una lista de los valores diferentes de ceros en orden arbitrario. La lista consiste de dos arreglos de enteros de i y j, y un arreglo real x de longitud igual al número de valores en la matriz. Por ejemplo la matriz:

$$A = \begin{pmatrix} 4.5 & 0 & 3.2 & 0 \\ 3.1 & 2.9 & 0 & 0.9 \\ 0 & 1.7 & 3.0 & 0 \\ 3.5 & 0.4 & 0 & 1.0 \end{pmatrix} \tag{2.23}$$

Una estructura de datos de base cero para una matriz de $m \times n$ contiene índices de renglón y columna en un rango de 0 a $m-1$ y $n-1$, respectivamente. Una estructura de datos de base uno tiene índices renglón y columna que inician en uno. En la Figura 2.15 se presenta la matriz (2.23) en una forma triple de base 0.

$$\text{int} \quad i[] = \{2, 1, 3, 0, 1, 3, 3, 1, 0, 2\}$$

$$\text{int} \quad j[] = \{2, 0, 3, 2, 1, 0, 1, 3, 0, 1\}$$

$$\text{double } x[] = \{3.0, 3.1, 1.0, 3.2, 2.9, 3.5, 0.4, 0.9, 4.5, 1.7\}$$

Figura 2.15: Forma triple base 0

La forma triple es sencilla de crear pero difícil de usar en la mayoría de los algoritmos de matrices dispersas.

La forma de columna condensada es más práctica y se usa en casi todas las funciones de CSparse. Una matriz dispersa de $m \times n$ que puede contener hasta $nzmax$ entradas se representa con un arreglo de enteros p de longitud $n+1$, un arreglo entero i de longitud $nzmax$, y un arreglo real x de longitud $nzmax$. Los índices de los renglones de las entradas en la columna j se almacenan desde $i[p[j]]$ hasta $i[p[j+1]-1]$, y los valores numéricos correspondientes en las mismas direcciones de x. El primer valor de $p[0]$ siempre es cero, y $p[n] \leq nzmax$ es el número de entradas reales en la matriz. La matriz (2.23) se representa en columna condensada en la Figura 2.16.

int $i[] = \{0, 3, 6, 8, 10\}$

int $j[] = \{0, 1, 3, 1, 2, 3, 0, 2, 1, 3\}$

double $x[] = \{4.5, 3.1, 3.5, 2.9, 1.7, 0.4, 3.2, 3.0, 0.9, 1.0\}$

Figure 2.16: Forma columna condensada

Para la estructura de datos de columna condensada se requiere que los índices de renglón en cada columna aparezcan en un orden ascendente, y no pueda haber entradas ceros.

El arreglo p contiene los apuntadores columna para la forma de columna condensada (de tamaño $n+1$) o los índices columna para la forma triple (de tamaño $nzmax$). La matriz se encuentra en forma columna condensada si nz es negativo.

2.4.2. Multiplicación de matrices dispersas

Ya que las matrices están almacenadas en forma columna condensada en CSparse, la multiplicación de matrices $C = A \times B$, donde C es de $m \times n$, A es de $m \times k$ y B es de $k \times n$, se debe accesar A y B por columna y crear una columna a la vez de C. Si C_{*j} y B_{*j} denota a la columna j de C y B, entonces $C_{*j} = A \times B_{*j}$. Dividiendo A en sus k columnas y B_{*j} en sus k entradas individuales:

$$C_{*j} = \begin{bmatrix} A_{*1} & \dots & A_{*k} \end{bmatrix} \begin{bmatrix} b_{1j} \\ \vdots \\ b_{kj} \end{bmatrix} = \sum_{i=1}^{k} A_{*i} b_{ij} \tag{2.24}$$

El patrón de no ceros de C es dado por el siguiente teorema:

Teorema 1 ([Gilbert 1994]) El patrón de no ceros de C_{*j} es la unión definida del patrón de no ceros de A_{*i} para toda i para la cual b_{ij} es no cero. Si C_j, A_i y B_j denotan el grupo de índices renglón de entradas no cero en C_{*j}, A_{*i} y B_{*j}, entonces

$$C_j = \bigcup_{i \in B_j} A_i. \tag{2.25}$$

El algoritmo de multiplicación de matrices debe computar a C_{*j} y C_j, un vector x denso se usa para construir C_{*j}. El grupo C_j se almacena directamente en C, pero otro vector de trabajo w se necesita para determinar si un índice renglón i esta ya en el grupo. El vector w empieza a liberarse, cuando se calcula la columna j, $w[i] < j+1$ denotará un índice renglón i que aún no está en C_j. Cuando i se inserta en C_j, $w[i]$ se cambia a $j+1$. La función sc_scatter calcula una iteración de (2.24) y (2.25) para un valor individual i usando una operación scatter para copiar un vector disperso en uno denso.

La función de multiplicación de matrices cs_multiply primero ubica los espacios de trabajo w, x y la matriz de salida C. Enseguida itera sobre cada columna j de resultado C. después de una serie de operaciones scatter, el vector denso x es convertido en un vector disperso (una columna de C). Debido a que el número de no ceros en C es desconocido al inicio se incrementa su tamaño según se necesite.

Cuando cs_multiply termina, la matriz C es redimensionada al tamaño real de entradas que contiene y el espacio de trabajo es liberado. La función cs_scatter calcula $x = x + beta \times A(:, j)$ y acumula el patrón de no ceros de x en $C- > i$ comenzando en la posición nz. El nuevo valor de nz se regresa y el índice de renglón i está en el patrón de x si $w[i]$ es igual a la marca.

2.4.3. Suma de matrices dispersas

La función cs_add realiza suma de matrices, $C = \alpha A + \beta B$. La suma de matrices puede ser escrita como la multiplicación de 2 matrices, tal como se muestra en la Ecuación (2.26).

$$C = \begin{pmatrix} A & B \end{pmatrix} \begin{pmatrix} \alpha I \\ \beta I \end{pmatrix} \tag{2.26}$$

Donde I es una matriz identidad de el tamaño apropiado. Aunque no es implementada de está manera [Timothy 2006], cs_add luce muy similar a cs_multiply por (2.26) difiere ligeramente; no se necesitan reubicaciones, y el ciclo $for\ p$ es remplazado por dos llamadas de cs_scatter.

2.5. Conclusiones

En este capítulo ha presentado de manera concisa las herramientas computacionales, matemáticas y numéricas que son utilizadas para el desarrollo de este trabajo de investigación. Se describen dos variantes de las técnicas de acercamiento rápido al estado estacionario periódico basadas en Diferenciación Númerica y la Matriz Exponencial Discreta.

Se mencionan las principales características de las arquitecturas actuales para el procesamiento en paralelo y se hace énfasis en la plataforma operativa OpenMP, que está asociada a la paralelización del método MED, debido a que permite que el programa sea portable a distintas computadoras con diferente número de núcleos y procesadores, sin la necesidad de realizar adaptaciones. También se describe de manera general el uso de la biblioteca CSparse para la implementación de técnicas de dispersidad.

Capítulo 3

Aplicación de Técnicas Numéricas y Computacionales a la técnica MED

3.1. Introducción

La exponencial de una matriz se puede calcular de varias maneras. Se han propuesto distintos métodos que involucran teoría de aproximación, ecuaciones diferenciales, valores característicos y el polinomio característico de la matriz [Moler y Loan 2003]. En este capítulo se presenta la paralelización del método Newton basado en la Matriz Exponencial Discreta, utilizando un esquema de memoria compartida, basado en la plataforma de OpenMP. También se incorporan técnicas de dispersidad con la librería CSparse [Timothy 2006], durante la identificación de la matriz transición necesaria para los métodos de Newton.

3.2. Método de Expansión Exponencial Discreta

El método Newton basado en la matriz exponencial discreta implementado emplea una extensión en series de Taylor en conjunto con el método de Escalamiento-Cuadratura (*Scaling-Squaring*, por sus siglas en inglés) para aproximar las matrices exponenciales [Moler y Loan 2003]. Este procedimiento es presentado en [Segundo y Medina 2010a] para la determinación de la solución en estado estacionario periódico. El diagrama de flujo del método de la MED implementado en esta investigación se presenta en la Figura 3.1:

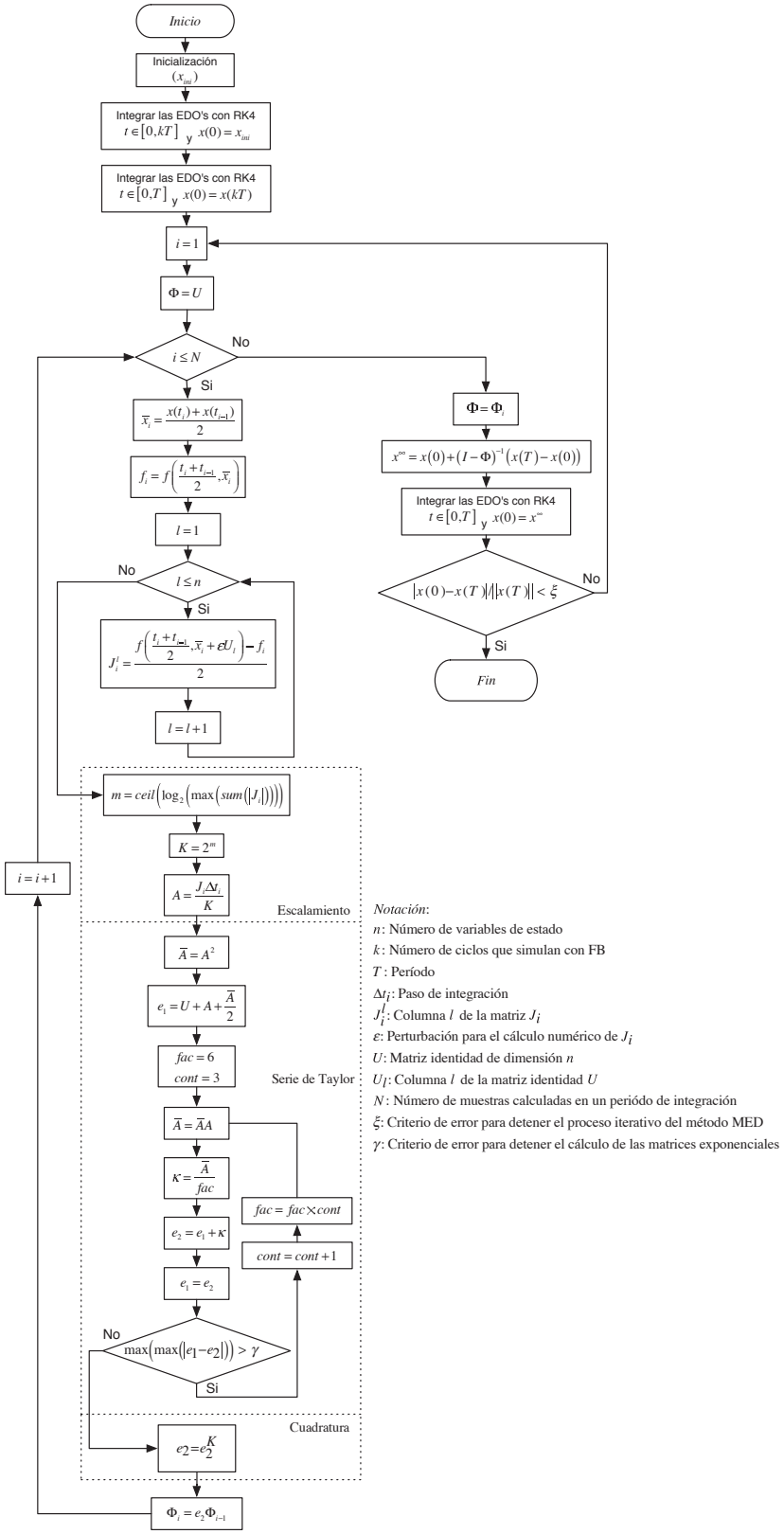


Figura 3.1: Diagrama de flujo método MED

El procedimiento para encontrar la solución periódica en estado estacionario basado en la Matriz Exponencial Discreta se puede resumir como sigue:

1. Se forma el sistema de ecuaciones diferenciales ordinarias que describen la dinámica de la red eléctrica.

2. El sistema de EDO's se resuelve para un número determinado de ciclos de integración iniciales kT con el método de integración de Runge-Kutta de cuarto orden (RK4) que se describe en el Apéndice A, se puede utilizar k periodos T, por ejemplo 3 si el sistema está bien amortiguado y 6 en caso contrario [Semlyen y Medina 1995].

3. Se construye el Jacobiano por columnas evaluando el sistema de EDO's con $\overline{x}_i$ y con un el valor perturbado $\overline{x}_i + \varepsilon$, donde $\varepsilon = 1 \times 10^{-8}$.

4. Se utiliza el proceso de escalamiento descrito en [Moler y Loan 2003] al Jacobiano construido en el punto 3.

5. Se calculan las matrices exponenciales con una aproximación basada en la expansión de series de Taylor.

6. Se utiliza el proceso de cuadratura a la matriz exponencial resultante de las series de Taylor, elevandola la matriz a la potencia K. Valor que se determinó en el paso 4.

7. Se obtiene una estimación de x^{∞}.

8. Verificar si satisface el criterio de convergencia especificado. Si la condición de convergencia se satisface el proceso concluye, en caso contrario se reinicia el proceso desde el punto 3 haciendo $x(0) = x^{\infty}$.

Es importante mencionar que el esfuerzo computacional para la determinación de Φ utilizando el método de la MED puede ser reducido si $\triangle t_i$ tiene un tamaño de dos veces el paso de integración original [Segundo y Medina 2010a].

Escalamiento y cuadratura

Se utiliza el método de escalamiento y cuadratura, para evitar los errores de redondeo y reducir el costo computacional de la expansión en la aproximación de series de Taylor, éstos dos aspectos se incrementan cuando la norma del Jacobiano se incrementa o cuando aumenta la dispersidad del Jacobiano. Ya que el Jacobiano es multiplicado por $\triangle t$, la norma del Jacobiano es pequeña; esto hace posible la aproximación de las matrices exponenciales en pocos términos de la series de Taylor.

3.2.0.1. Efecto del paso de integración en la presición de la solución en el método de la MED

El tamaño de paso de integración para la solución del conjunto de EDO's que modelan un sistema eléctrico, afecta directamente la precisión de la solución. Normalmente se desea emplear el paso de integración más grande posible para obtener una solución más rápida, pero no tan grande que introduzca errores apreciables de esta. Al utilizar los métodos de integración de paso fijo, es muy importante la selección del paso de integración adecuado.

Para ejemplificar ésto de manera más clara, considere la Figura 3.2, que presenta una red eléctrica trifásica de dos nodos, que cuenta con dos líneas de transmisión, un generador y un compensador estático de potencia reactiva o STATCOM (por sus siglas en inglés).

El conjunto de 17 EDO's que describen la dinámica de la red mostrada en la Figura 3.2 se presenta en el Apéndice B.

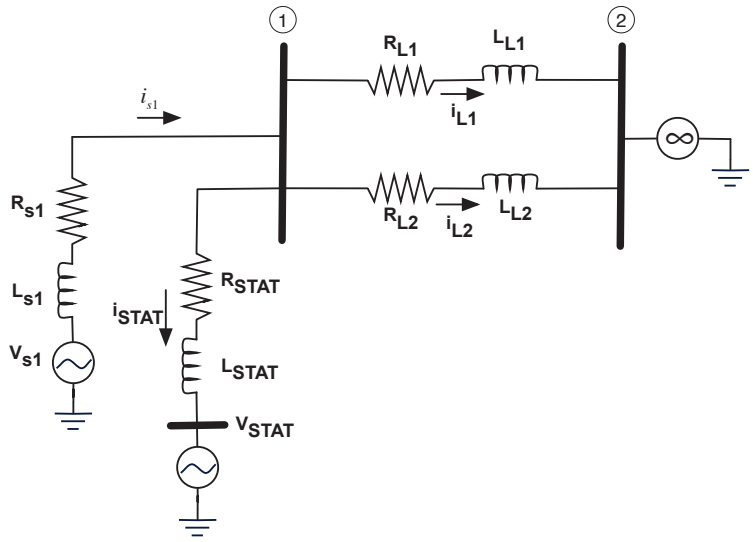


Figura 3.2: Red de dos nodos con un STATCOM

En la Figura 3.3 se muestra la forma de onda del voltaje en el capacitor del STATCOM con tres pasos de integración diferentes y utilizando el método de integración RK4. La simulación de Simulink se realiza con un método de integración de paso variable (ode15s), especificando un error absoluto y relativo de 1×10^{-12}, se puede observar fácilmente como se deteriora la solución a medida que se aumenta el paso de integración y la necesidad de utilizar pasos suficientemente pequeños para tener una solución lo más cercana a la obtenida con Simulink, de la Figura 3.3 se observa que la más cercana a la de Simulink es la obtenida con un paso de

integración de $8.13\mu s$, aunque con este paso de integración la solución no tiene la precisión de 1×10^{-12} que se obtiene con Simulink, por lo tanto, los errores de convergencia en la iteración del método no son estrictamente el error de la solución de estado estacionario del sistema de EDO's.

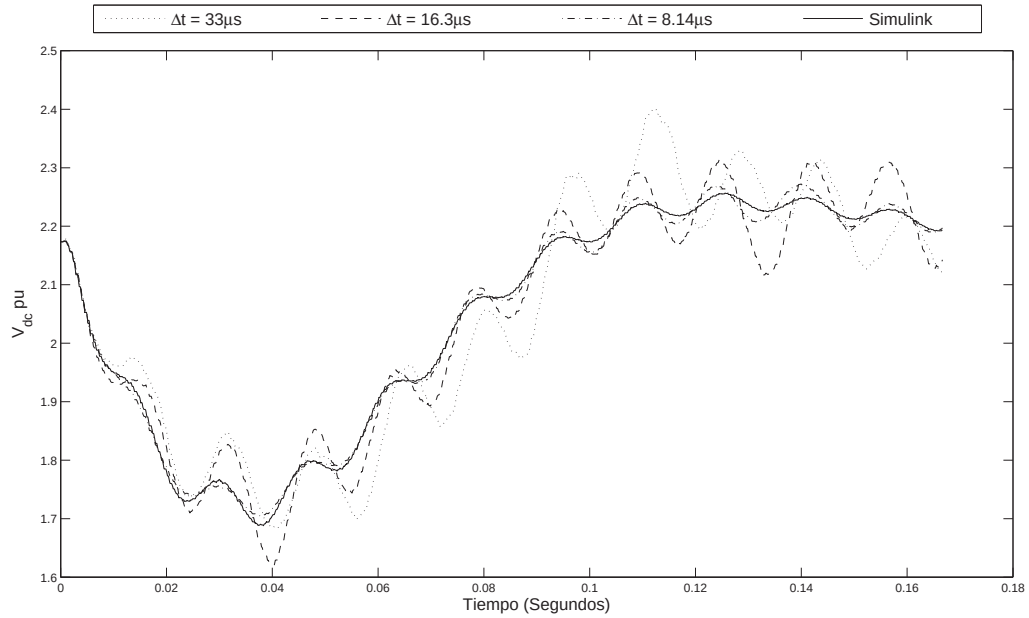


Figura 3.3: Efecto del paso de integración

En los resultados mostrados en la Figura 3.3 se observa que la precisión de la solución en EEP obtenida será función del paso de integración, ya que el método MED se basa en aproximar el sistema no lineal por sistemas lineales sucesivos para cada paso de integración; éste influirá directamente en la convergencia del método. Para ilustrar mejor ésto, se calcula la solución en EEP con el método de la MED con diferentes paso de integración y se muestra el comportamiento de la convergencia en la Tabla 3.1.

Tabla 3.1: Convergencia del método MED con diferentes pasos de integración

Iteraciones	$\triangle t = 33\mu s$	$\triangle t = 16.3\mu s$	$\triangle t = 8.14\mu s$	$\triangle t = 4.06\mu s$
1	1.625236E-03	1.288128E-03	1.285910E-03	4.489040E-04
2	2.088551E-04	1.442802E-05	1.964661E-05	2.285032E-06
3	4.157119E-05	1.512656E-07	7.526763E-08	7.178697E-09
4	4.979967E-06	4.388194E-10		
5	1.197071E-06			
6	1.453639E-07			
7	3.469814E-08			

De la Tabla 3.1 se observa que entre menor sea el paso de integración, el número de iteraciones necesarias del método MED para satisfacer el criterio de convergencia de 1×10^{-7}

es menor. Por lo tanto, el paso de integración utilizado para los casos de estudio reportados en las sub-secciones siguientes es de $8.13\mu s$. Además, con este paso de integración, se obtiene un mejor característica de convergencia del método MED.

3.2.1. Criterio de convergencia

En la contribución hecha por [Semlyen y Medina 1995] se ha adoptado una tolerancia para convergencia de $1 \times 10^{-10} p.u.$, sin embargo, este valor depende en particular de las bases de potencia y voltaje para cada red electrica analizada. El error es calculado como la máxima diferencia absoluta entre el vector de estados de dos ciclos sucesivos. Una vez que se tiene el error, se compara con el índice de convergencia y si el error es menor que este índice se considera que se está en el Ciclo Límite.

Se determina la solución en EEP de la red eléctrica trifásica mostrado en la Figura 3.2 con el método FB utiliza el método de integración RK4 y con un paso de integración de $8.13\mu s$, con el propósito de observar el impacto en las formas de onda y su espectro armónico ante diferentes valores de ξ.

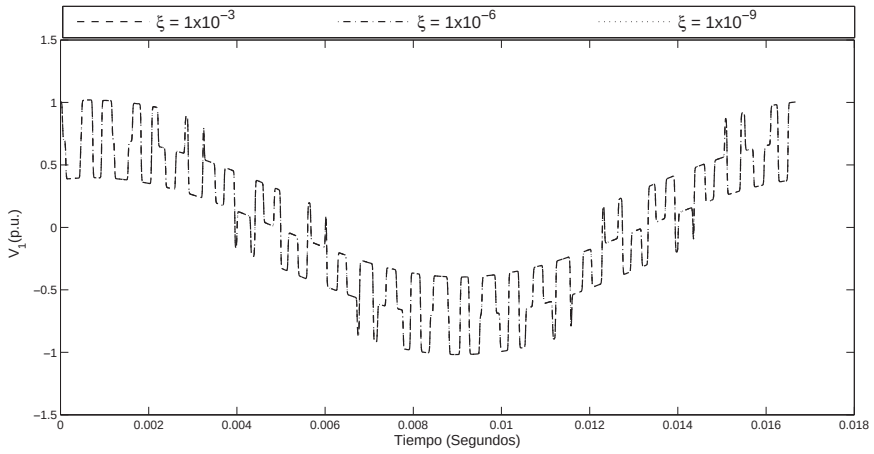


Figura 3.4: Formas de onda del voltaje V_1 con diferente índice de convergencia

Es claro que la solución en estado estacionario obtenida será función del paso de integración en el caso de los métodos de paso fijo. De acuerdo con esto, se sugiere adoptar que el valor de ξ sea escogido en función del estudio que se pretende realizar después de obtener la solución en estado estacionario. Por ejemplo, una precisión de $1 \times 10^{-7} p.u.$ pudiera ser suficiente para cuantificar la distorsión armónica en redes de potencia, atendiendo a sus valores base; así como para inicializar los programas tipo EMTP (*ElectroMagnetic Transients Program*) y

para la determinación de especificaciones prácticas de circuitos electrónicos. Sin embargo, se requieren soluciones más rigurosas y precisas en estado estacionario para la determinación de diagramas de bifurcación [Parker y Chua 1989].

En la Figura 3.4 se presenta un periodo completo del voltaje en el nodo 1 con diferentes índices de convergencia, se observa que no hay diferencia en las formas de onda en EEP determinadas por FB con los diferentes índices de convergencia propuestos.

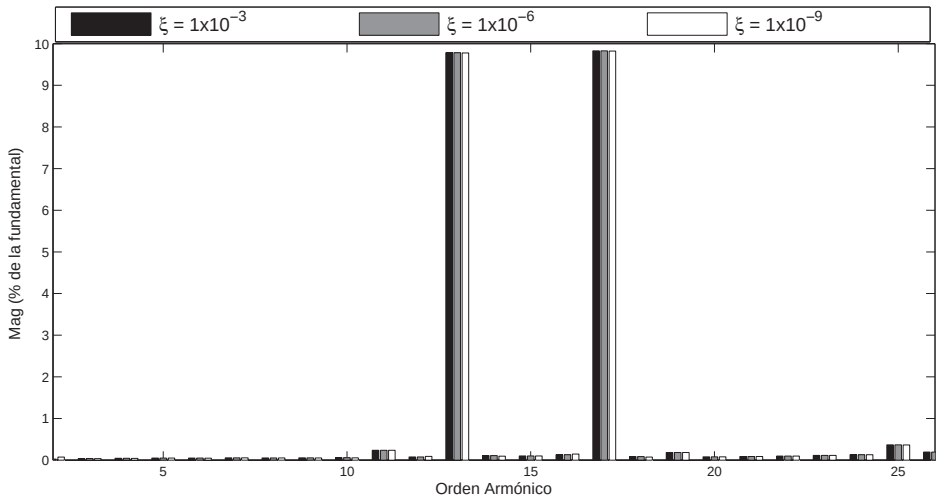


Figura 3.5: Contenido armónico de la Figura 3.4

El contenido armónico de las formas obtenidas con los diferentes índices de convergencia se presentan en la Figura 3.5, se sugiere que el valor de ξ para determinar si la solución está en el Ciclo Límite puede ser mayor a la propuesta en [Semlyen y Medina 1995], ya que utilizar una tolerancia para convergencia de $1 \times 10^{-10} p.u.$ resulta computacionalmente costoso. Por ésto, se propone que un buen valor para ξ es de $1 \times 10^{-7} p.u.$, ya que es una precisión alta, en el caso en que sólo quisiera determinarse el contenido armónico de las formas de onda en estado estacionario.

3.2.2. Generación de EDO's

La información requerida por el software desarrollado en [Ramos 2007] para la generación de las Ecuaciones Diferenciales Ordinarias que describen la dinámica de varios de los casos de estudio reportados en esta tesis, se introducen a través de un archivo de datos con formato ASCII, el cual puede ser generado por cualquier editor de texto. La Tabla 3.2 muestra la estructura que debe tener el archivo de datos.

Tabla 3.2: Estructura del archivo de entrada para la el software de [Ramos 2007]

1	Renglón de parámetros generales del sistema
2	Líneas de transmisión monofásica
3	Bancos de capacitores
4	Ramas magnetizantes
5	Generadores
6	Hornos de Arco

El rengón de parámetros generales del sistema contiene una lista con 2 parámetros enteros, el primero representa el número de nodos del sistema y el segundo especifica el número de elementos del sistema. Los renglones siguientes están ligados con la información de cada uno de los elementos que forman el sistema eléctrico de potencia. A continuación se muestra como se debe de introducir los parámetros de cada uno de los elementos.

Tabla 3.3: Parámetros de los elementos para la Generación de EDO's

Línea de transmisión									
Identificador	Nodo de envío	Nodo de recepción	R	L					
Banco de capacitores									
Identificador	Nodo de conexión	Nodo de referencia	C						
Rama magnetizante									
Identificador	Nodo de conexión	Nodo de referencia	R	Exp					
Generador									
Identificador	Nodo de conexión	Nodo de referencia	Amplitud máxima	Ángulo de fase					
Hornos de arco									
Identificador	Nodo de conexión	Nodo de referencia	I	k1	k2	k3	n	m	Condición inicial para el radio

Los modelos en el dominio del tiempo que describen el comportamiento de los elementos monofásicos de la planta de potencia, tales como línea de transmisión, generadores, rama magnetizante del transformador, banco de capacitores, hornos de arco, etc. considerados en los distintos casos de estudio se describen en el Apéndice C.

3.3. Técnicas de procesamiento en paralelo

El método de la Matriz Exponencial Discreta durante el proceso de solución implica operaciones matriciales; suma, resta, potencia y multiplicación. Éstas operaciones matriciales son altamente paralelizables, en la Sección 2.3.5 se presentan varias plataformas operativas de procesamiento en paralelo que pueden ser aplicables. El beneficio de utilizar OpenMP radica en la comodidad relativa de código de paralelización hecho posible para la arquitectura de memoria compartida, para tener un estándar que asegure la portabilidad. Por estas razones, la plataforma que se eligió para el desarrollo de esta tesis es OpenMP, descrito en la Sección 2.3.5.4.

Durante la implementación del método de la Matriz Exponencial Discreta se identificó que un proceso demandante durante el proceso de solución, es la potencia que se utiliza para cuadrar la matriz resultante de la expansión en series de Taylor (Figura 3.1); por este motivo en este proceso se incorporó el procesamiento paralelo basado en OpenMP.

Para realizar la paralelización del método de la matriz exponencial discreta se utilizó el lenguaje de programación C. Los códigos fueron ejecutados en una computadora Intel Xeon con dos procesadores cada uno de los cuales cuenta con 4 núcleos con una velocidad de 2.0 GHz, el sistema operativo utilizado por la computadora es Xubuntu, que es un sistema basado en Unix.

3.3.1. Propuesta de la paralelización de la multiplicación matricial

La forma secuencial de multiplicación matricial se realiza como se ilustra en la Figura 3.6, un sólo procesador calcula todos lo elementos de la matriz C; éste proceso resulta computacionalmente demandante en matrices de mediana y gran escala.

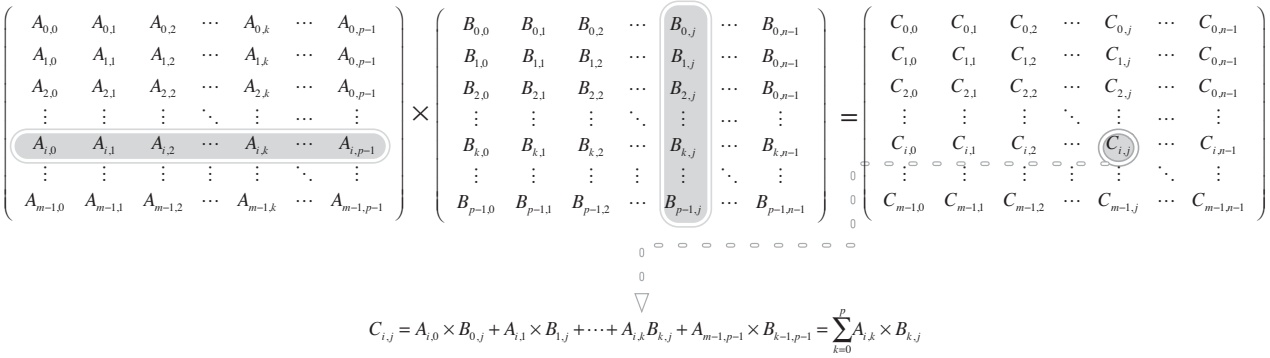


Figura 3.6: Multiplicación secuencial de $A_{m\times p}B_{p\times n} = C_{m\times n}$

La Tabla 3.4 muestra un fragmento del código 3.4 implementado en lenguaje C para el cálculo de la multiplicación matricial secuencial que se describe en la Figura 3.6.

Tabla 3.4: Multiplicación secuencial

```
double f(int i, int j){
        double resul=0;
        int i;
        for (k=0; k<p; k++)
                resul += A[i+k*p]*B[k+j*p];
        return resul;
}
void multiplica(){
        for (i=0; i<var_est; i++){
                for (j=0; j<var_est; j++) {
                        C[i+j*var_est]=f(i,j);
                }
        }
}
```

La Figura 3.7 muestra la propuesta de paralización de la multiplicación matricial, que se utilizo para hacer más eficiente el método basado en la matriz exponencial discreta.

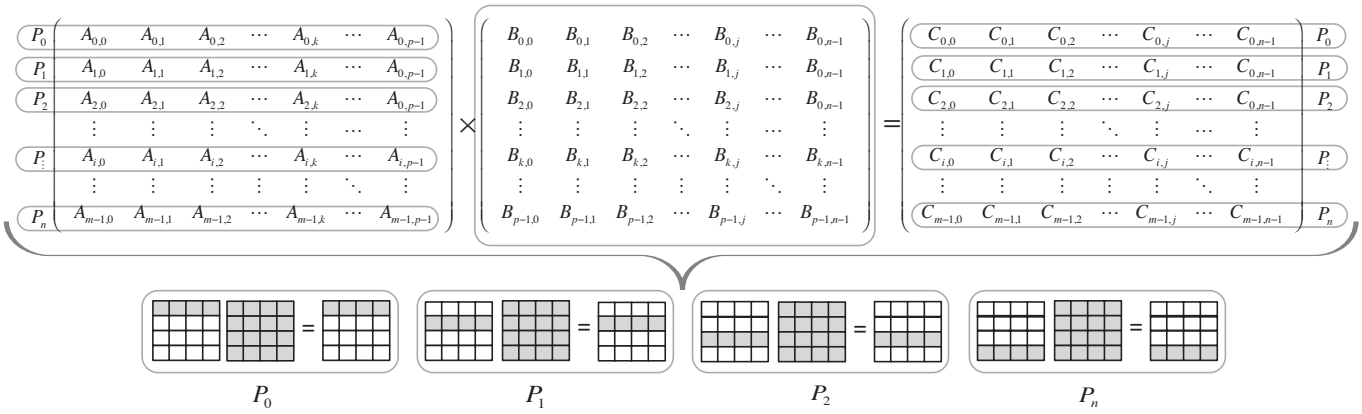


Figura 3.7: Multiplicación matricial paralela de $A_{m\times p}B_{p\times n} = C_{m\times n}$

Observe, de la Figura 3.7, que cada fila de la matriz A se divide entre el número de procesadores. Los elementos de procesos se dividirán en una cantidad igual a la razón entre el número de filas de la matriz A y el número de elementos de procesadores. Además, cada procesador tiene acceso a la matriz B, para así realizar la multiplicación matricial de forma paralela. Una vez realizada la operación, cada procesador se encarga de escribir el resultado en la matriz C, de forma sincronizada, es decir, cada resultado de salida es impreso en la matriz C por un procesador, sin ninguna interferencia de otros procesadores.

En la Tabla 3.5 se muestra una parte del código utilizado para el cálculo en paralelo de la multiplicación matricial.

Tabla 3.5: Multiplicación matricial paralela

```
double f(int i, int j){
        double resul=0;
        int i;
        for (k=0; k<p; k++)
                resul += A[i+k*p]*B[k+j*p];
        return resul;
}
void multiplica(){
#pragma omp parallel for private(i,j)
        for (i=0; i<var_est; i++){
                for (j=0; j<var_est; j++) {
                        C[i+j*var_est]=f(i,j);
                }
        }
}
```

Comparando los códigos de las Tablas 3.4 y 3.5, se observa que sólo difieren en una línea, que es donde se inserta el pragma de información que utiliza el compilador para asignar el trabajo a cada procesador. OpenMP aporta una gran ventaja computacional a este proceso. Una vez desarrollado el algoritmo y código es fácil paralelizarlo. Este algoritmo permite agilizar el proceso del producto matricial hasta 7 veces; para ejemplificar esto de manera más clara se muestra en la Figura 3.8 el tiempo de ejecución de la multiplicación de dos matrices de tamaño de 500×500.

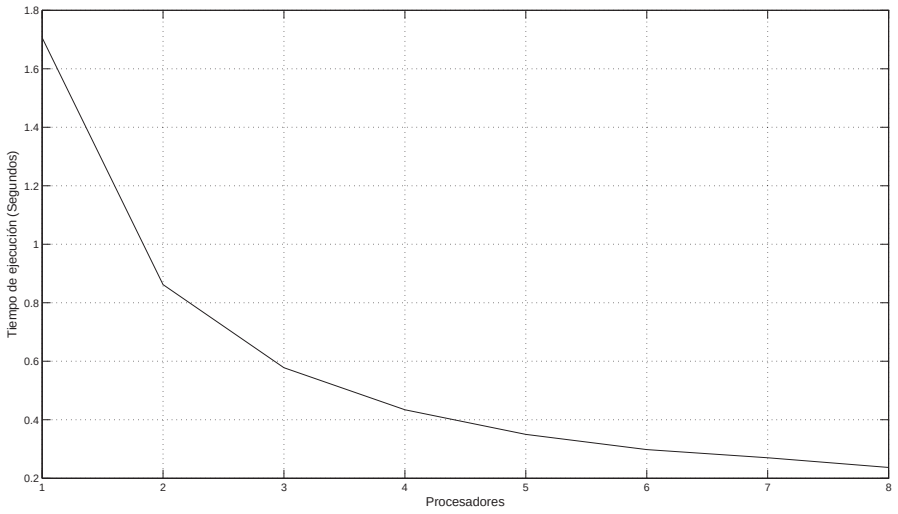


Figura 3.8: Tiempo de ejecución de la multiplicación de dos matrices de tamaño 500×500

3.3.2. Potencia de una matriz

La potencia de una matriz se puede realizar como una sucesión de multiplicaciones matriciales, ya que el producto matricial se paraleliza de acuerdo al procedimiento descrito en

la sección anterior. Se obtiene así un ahorro importante en el tiempo de cómputo requerido para realizar la cuadratura que necesita el método de la Matriz Exponencial Discreta. El cálculo de $e_2 = e_2^K$ donde $K = 2^m$ se puede realizar como se muestra en la Figura 3.9.

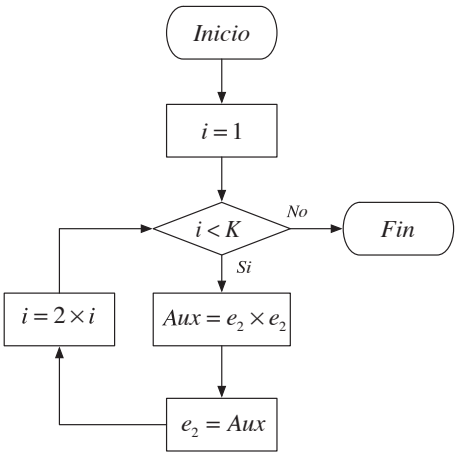


Figura 3.9: Diagrama del proceso $e_2 = e_2^K$

Es importante mencionar que la matriz resultante de la expansión en series de Taylor es altamente dispersa, pero pierde su dispersidad cuando se eleva a una potencia K. Por este motivo no es eficiente realizar las multiplicaciones matriciales que involucra y el proceso de cuadratura de forma dispersa y se opto por realizar estos productos con esta implementación paralela, el proceso para el cálculo de e_2 se incrementa 7.5 veces; para mostrar esto de manera más clara se presenta en la Figura 3.10 el tiempo de ejecución utilizando diferente número de procesadores de la potencia de una matriz cuadrada.

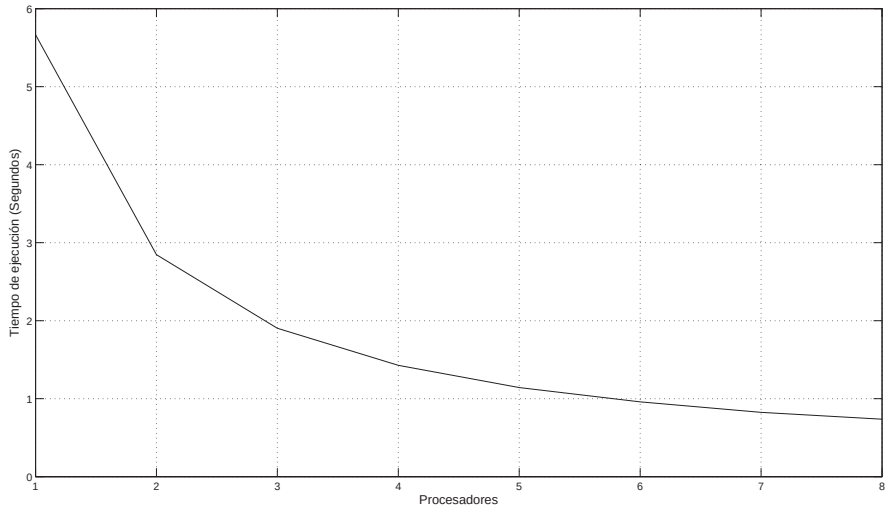


Figura 3.10: Tiempo de ejecución de una matriz de tamaño 500×500 elevada a la potencia 32

3.4. Técnicas de dispersidad

Las incorporación de las técnicas de dispersidad en el método de la Matriz Exponencial Discreta se realiza en la expansión de series de Taylor, que se muestra en la Figura 3.1, debido a que una propiedad que tiene el Jacobiano es que tiene pocos elementos diferentes de cero, es decir, que es altamente disperso. Esto quiere decir que el Jacobiano conserva su característica de dispersidad, por esta razón es importante realizar las operaciones matriciales implicadas en el cálculo de la serie de Taylor con técnicas de dispersidad.

La librería "cs.h" [Timothy 2006] permite realizar operaciones de forma dispersa, pero las funciones de esta librería presentadas en el Apéndice E requieren de una matriz dispersa en una estructura de datos de la forma cs como entrada.

3.5. Conclusiones

En este capítulo se describió el desarrollo, e implementación del método de la Matriz Exponencial Discreta, para el análisis en el dominio del tiempo de sistemas eléctricos lineales, no lineales y variantes en el tiempo. Con el objeto de reducir el tiempo de computo del proceso de solución de esté método se incorporaron técnicas de dispersidad y procesamiento en paralelo, donde se identificaron procesos con altos costos computacionales.

Adicionalmente, en este capítulo se presenta un análisis del criterio de convergencia adecuado para determinar el contenido armónico de las formas de onda obtenidas en estado estacionario. Se muestra que la convergencia del método de la Matriz Exponencial Discreta depende del paso de integración, entre más pequeño sea el paso de integración, la convergencia del método se comporta de forma cuadrática. En principio, esto no representa ningún inconveniente, ya que para los errores de convergencia considerados dentro de un criterio de error de 1×10^{-7}, se necesitan en general pasos de integración de aproximadamente $8.14\mu s$. Por otro lado, si únicamente se está interesado en las formas de onda y su contenido armónico, pero no en su precisión real, pasos de integración de varios microsegundos pueden ser utilizados y en ese caso el MED no presenta ventajas sobre DN.

Capítulo 4

Casos de Estudio

4.1. Introducción

En este capítulo se presenta la solución periódica en estado estacionario obtenida con los métodos de DN y MED, para 5 casos de estudio monofásicos. Estos casos de estudio consisten en redes eléctricas con elementos lineales y no lineales. En estos casos se presenta:

- La comparación entre las iteraciones necesarias que ocupa cada uno de los métodos de DN y MED, para la determinación de la solución en EEP.
- El contenido armónico y la distorsión armónica total ($\%THD$, por sus siglas en inglés) de la forma de onda obtenida del sistema en EEP.
- La Eficiencia Relativa obtenida con la incorporación de procesamiento en paralelo usando la plataforma de OpenMP y técnicas de dispersidad.

Los elementos que conforman los casos de estudio son líneas de transmisión, generadores, banco de capacitores, ramas magnetizantes y hornos de arco. Estos elementos se han modelado como se reporta en [Ramos 2007], mismos que se describen en el Apéndice C.

En los casos de estudio reportados a continuación se emplea un criterio de convergencia de 1×10^{-7}; el paso de integración utilizado es $\triangle\, t = 8.14\,\mu s$.

Las ecuaciones que describen la dinámica de los casos de estudio fueron obtenidas con la herramienta de simulación desarrollada por [Ramos 2007].

4.2. Sistema de 3 nodos

Este caso de estudio consiste en un sistema eléctrico monofásico que cuenta con tres líneas de transmisión, tres bancos de capacitores, un generador y dos ramas magnetizantes

de los transformadores de potencia en paralelo con dos capacitores. La Figura 4.1 muestra el diagrama del sistema eléctrico a ser analizado en este caso de estudio.

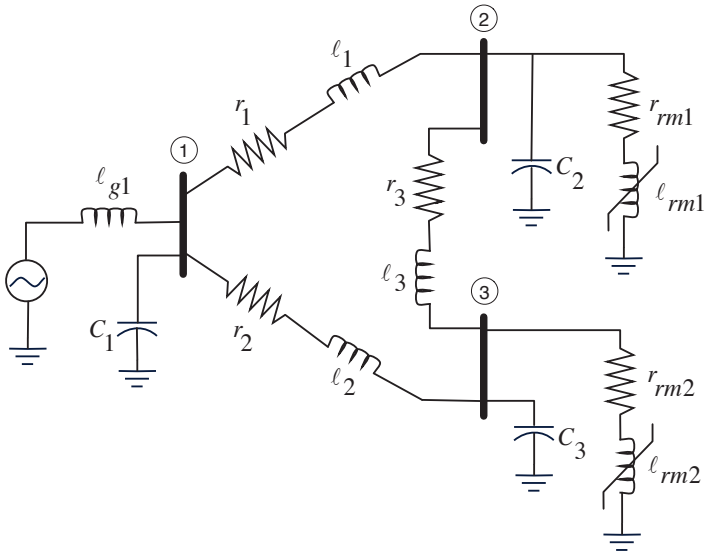


Figura 4.1: Caso de estudio 1: Sistema eléctrico de 3 nodos monofásico

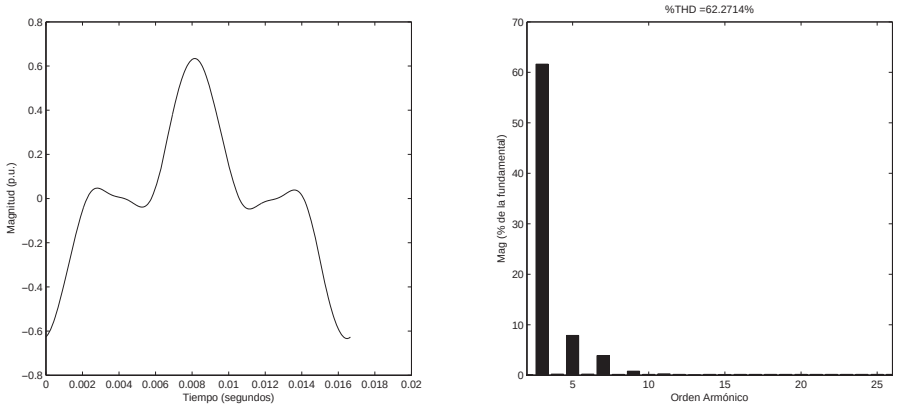
La Tabla D.1 del Apéndice D muestra el archivo de texto requerido por la herramienta de simulación que se reporta en [Ramos 2007] con el objeto de generar las EDO's que describen la dinámica del sistema mostrado en la Figura 4.1. Este sistema eléctrico es representado por un conjunto de 9 EDO's. En la Tabla 4.1 se muestra la comparación del proceso de convergencia hacia la solución EEP obtenida con los métodos de DN y MED. El método de FB requiere de 53 periodos completos de integración para determinar la solución en EEP, mientras que el método de DN requiere de solo 44 periodos completos de integración, que están asociados con 4 aplicaciones de la técnica de DN, mientras que el método de MED requiere de 3 aplicaciones (7 periodos).

Tabla 4.1: Convergencia de los métodos DN y MED

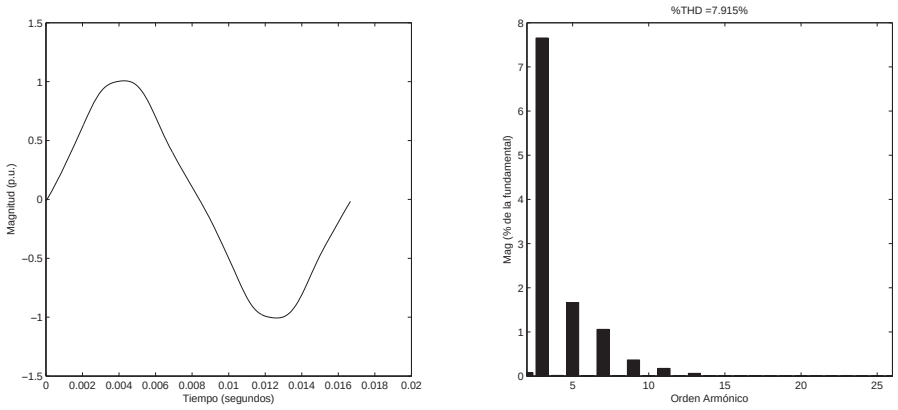
Iteraciones	DN	MED
1	3.835131E-02	9.304537E-04
2	9.311866E-04	8.076168E-07
3	2.562706E-07	5.592196E-08
4	3.111968E-14	

En la Figura 4.2 se muestran las formas de onda en estado estacionario de dos variables de estado asociadas al sistema eléctrico mostrado en la Figura 4.1. La Figura 4.2a muestra un

ciclo completo y el contenido armónico de la corriente en la línea de transmisión conectada entre el nodo 1 al nodo 2. En la Figura 4.2b se muestra el comportamiento en el tiempo del flujo en la rama magnetizante conectada al nodo 3 del sistema eléctrico. Adicionalmente, en las Figuras 4.2 se muestra el $\%THD$ de cada una de las formas de onda en EEP.



(a) Forma de onda de la corriente en la Línea 1-2



(b) Flujo en la rama magnetizante 2

Figura 4.2: Formas de onda de algunas variables de estado y su contenido armónico

Adicionalmente, se realizaron pruebas de Eficiencia Relativa aplicando la plataforma de OpenMP al método MED. Sin embargo, para este sistema no se aprecia la reducción de tiempo de cómputo debido a que el tiempo ahorrado en el proceso de cómputo es contrarrestado por la sincronización interna de los hilos dentro de la plataforma de OpenMP, llegando a ser ineficiente la paralelización del método MED para este caso. Esta situación se ilustra en la Figura 4.3.

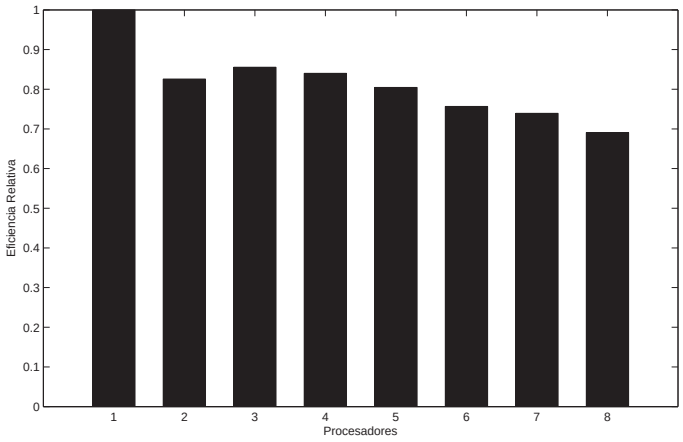


Figura 4.3: Eficiencia relativa con el uso de 1-8 procesadores para el Caso de Estudio 4.2

4.3. Sistema de IEEE de 14 nodos modificado

El sistema eléctrico de análisis es el sistema de 14 nodos presentado en la Figura 4.4. Este sistema está conformado por 20 líneas de transmisión, 2 generadores y 14 bancos de capacitores [ee.washington.edu]. Este sistema fue modificado por medio de la incorporación de dos ramas magnetizantes conectadas en los nodos 2 y 10 del sistema eléctrico.

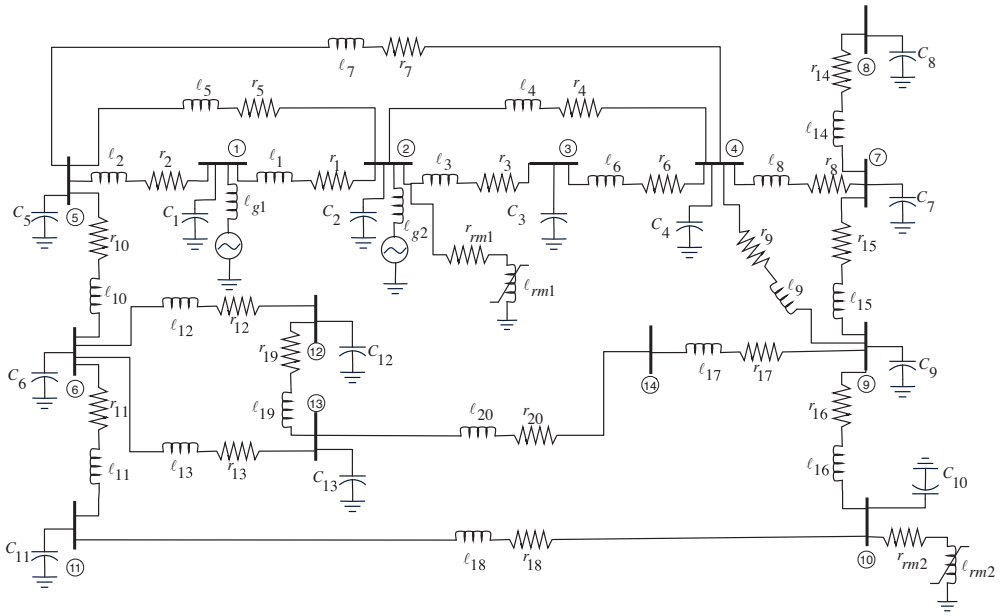


Figura 4.4: Caso de estudio 2: Sistema eléctrico de 14 nodos monófasico

La Tabla D.2 muestra el archivo de texto que requiere el generador de ecuaciones diferenciales, el sistema eléctrico de la Figura 4.4; es modelado por medio de un conjunto de 38 EDO's. En la Tabla 4.2 la convergencia de la solución al Ciclo Límite de los métodos de DN y MED, el método de DN requiere de 160 periodos (4 iteraciones) para llegar al Ciclo Límite, mientras que el método de FB requiere 1282 ciclos completos de integración para llegar a solución en EEP. El método MED requiere de 3 iteraciones (7 periodos), para lograr satisfacer el criterio de convergencia especificado.

Tabla 4.2: Convergencia de los métodos DN y MED

Iteraciones	DN	MED
1	3.835131E-02	9.304537E-04
2	9.311866E-04	8.076168E-07
3	2.562706E-07	5.592196E-08
4	3.111968E-14	

En las Figuras 4.5 se muestran las formas de onda elegidas en EEP, la Figura 4.5a está asociada voltaje del nodo 4 y la Figura 4.5b presenta un ciclo completo del comportamiento del flujo en la rama mágnetizante conectada al nodo 10.

La distorsión en las formas de onda se debe a la presencia de las ramas magnetizantes del transformador, cuyo punto de operación se encuentra en la región de saturación. En la Figura 4.5a se observa un elevado contenido armónico individual; hasta el treceavo armónico, así como de THD; ambos por encima de la norma IEEE 519 vigente.

Para éste caso también se hicieron pruebas de Eficiencia Relativa utilizando OpenMP al método MED. Sin embargo, para este sistema no se aprecia reducción de tiempo de cómputo debido a la misma razón que se explicó en el caso de estudio 4.2. La Figura 4.6 ilustra esta situación.

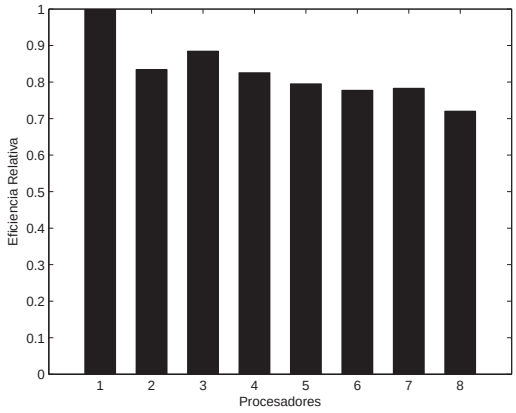
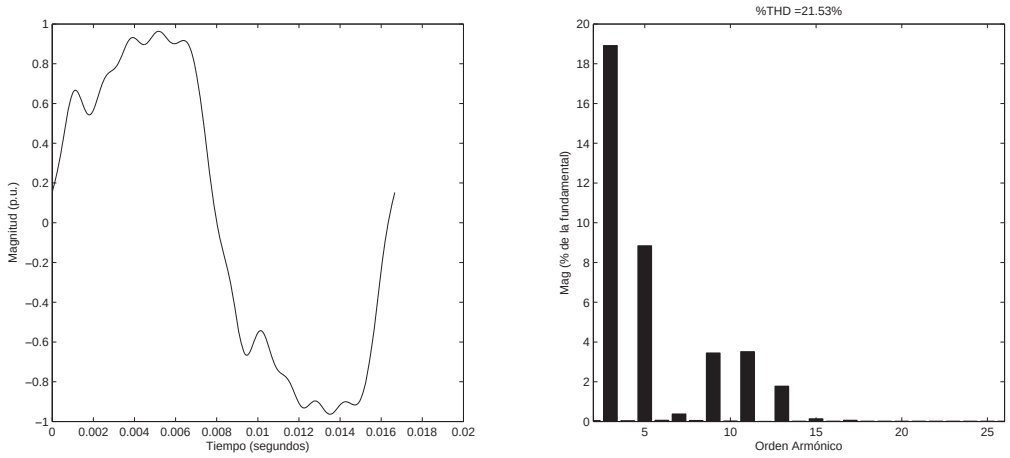
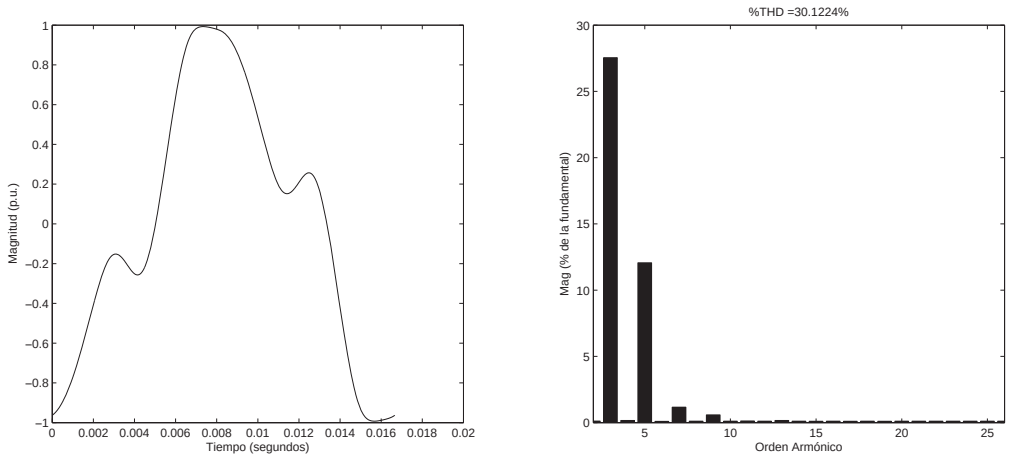


Figura 4.6: Eficiencia relativa con el uso de 1-8 procesadores para el Caso de Estudio 4.3



(a) Forma de onda del voltaje en el nodo 4



(b) Flujo en la rama magnetizante

Figura 4.5: Formas de onda de algunas variables de estado y su contenido armónico

4.4. Sistema de IEEE de 30 nodos modificado

El sistema eléctrico de análisis para este caso de estudio cuenta con 30 nodos y esta conformado por 41 líneas de transmisión, 30 bancos de capacitores y 6 generadores [ee.washington.edu]. Éste sistema es modificado con la incorporación de un horno de arco conectado al nodo 22. El comportamiento del sistema eléctrico se puede representar por medio de 79 EDO's.

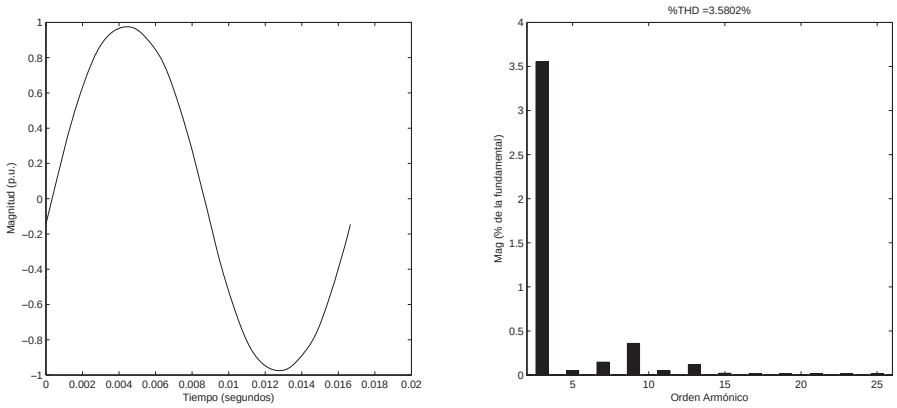
Para este Caso de Estudio el método de FB requiere de 1116 ciclos (periodos) para alcanzar la solución en EEP. De la Tabla 4.3 se observa que el método de DN le toma 324 ciclos o 29.03 % de aquellos requeridos por el método FB. El método de la MED requiere 8 ciclos y de las mismas iteraciones que el DN, convergiendo al Ciclo Límite estos dos métodos

con mayor precisión que el método de FB.

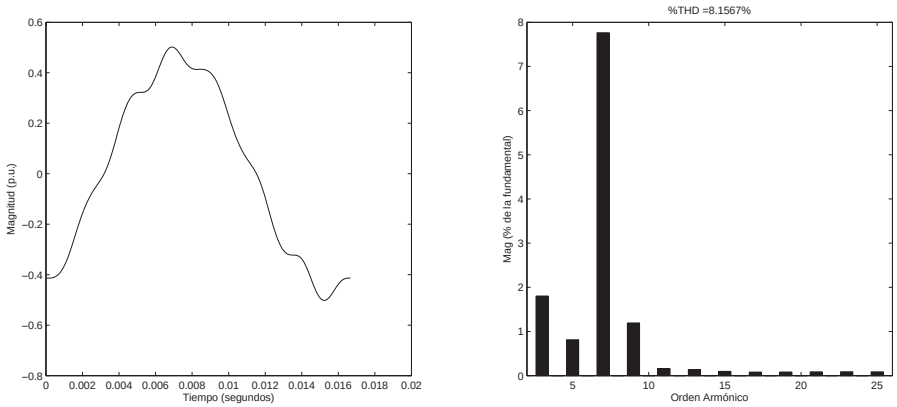
Tabla 4.3: Convergencia de los métodos DN y MED

Iteraciones	DN	MED
1	2.869147E-02	1.692495E-03
2	1.691892E-03	1.882806E-05
3	3.895404E-07	6.181053E-07
4	7.481469E-14	1.509641E-08

En las Figuras 4.7a y 4.7b, se muestra la evolución en el tiempo de algunas variables de estado representativas en EEP. En la 4.7a se muestra el voltaje en el nodo 29, el cual tiene un contenido armónico de 3.5 % de la fundamental para el tercer armónico; la $\%THD = 3.58\,\%$. La 4.7b muestra la forma de onda de la corriente que circula en la línea de transmisión que une a los nodos 12 y 16 del sistema. El contenido armónico de ésta corriente está asociado principalmente con el séptimo armónico con magnitud cercana al 8 % y tiene un $\%THD = 8.15\,\%$.



(a) Forma de onda del voltaje en el nodo 29



(b) Forma de onda de la corriente en la Línea 12-16

Figura 4.7: Formas de onda de algunas variables de estado y su contenido armónico

La Figura 4.8 muestra de forma gráfica la eficiencia relativa obtenida con el uso de 1 a 8 procesadores, se puede apreciar que la eficiencia relativa máxima es cercana a 3.5 y se obtiene utilizando 8 procesadores.

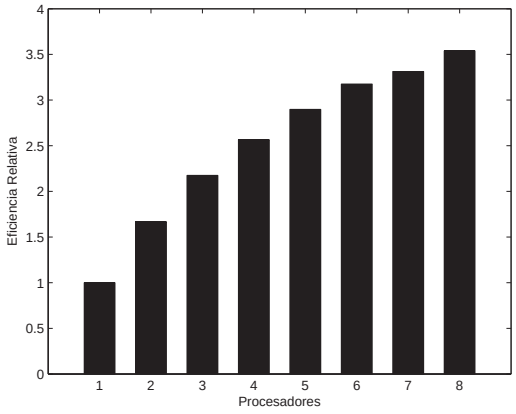


Figura 4.8: Eficiencia relativa con el uso de 1-8 procesadores para el Caso de Estudio 4.4

4.5. Sistema de IEEE de 57 nodos

En este caso de estudio, se presenta un sistema eléctrico de 57 nodos [ee.washington.edu], que consta de 78 líneas de transmisión, 57 bancos de capacitores, 7 generadores y 3 ramas magnetizantes, conectadas en los nodos 57, 23 y 9. Este sistema es modelado por medio de 145 EDO's.

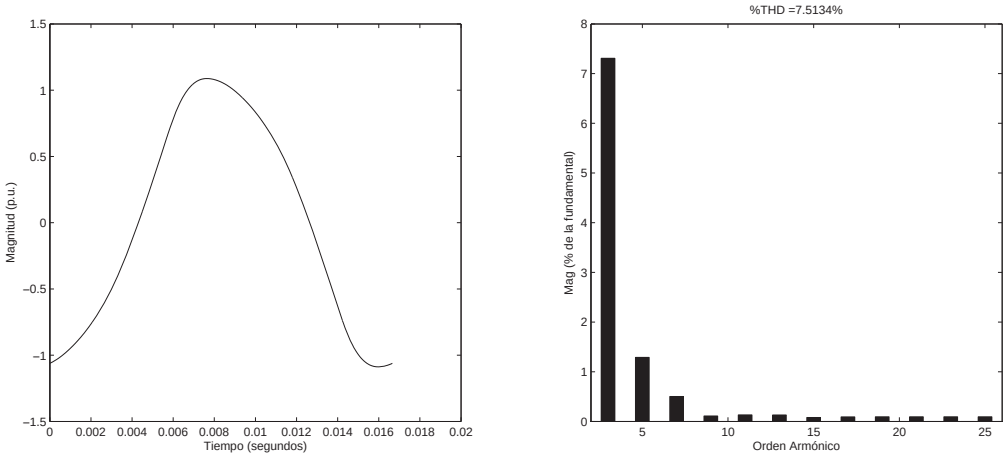
En la Tabla 4.4 se muestra el proceso de convergencia al EEP de la red de 57 nodos, se puede ver que el método de DN requiere de 4 iteraciones (584 ciclos), mientras que el método de la MED ocupa dos iteraciones más que DN y 10 periodos completos de integración; mientras que el método de FB requiere de 481 periodos completos de integración.

Tabla 4.4: Convergencia de los métodos DN y MED

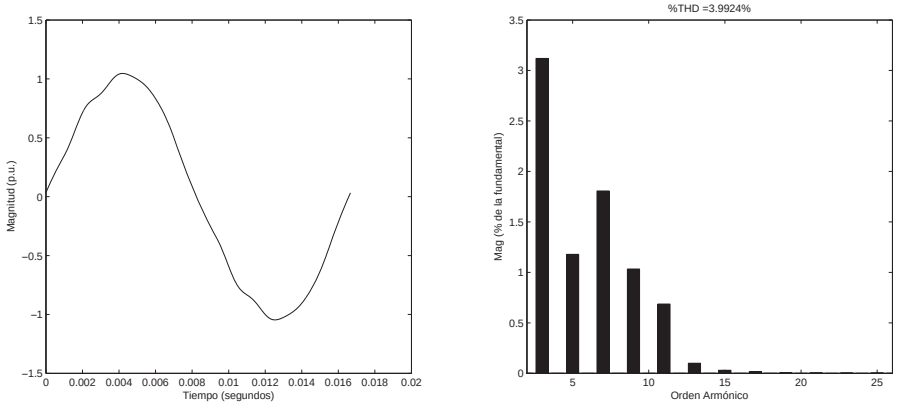
Iteraciones	DN	MED
1	1.265099E-01	2.742765E-03
2	2.750820E-03	7.672692E-05
3	1.072068E-05	1.507113E-05
4	2.087527E-11	2.910414E-06
5		5.102893E-07
6		8.216697E-08

En la Figura 4.9 se muestran formas de onda representativas de las variables de estado asociadas con el caso de estudio analizado. La forma de onda del voltaje en el nodo 9 se

presenta en la Figura 4.7a; también se presenta su contenido armónico, contiene un tercer armónico cercano al 7.5 % y un porcentaje de $THD = 7.51\,\%$. La Figura 4.7b muestra el comportamiento del flujo en la rama magnetizante conectada al nodo 57; además se muestra el espectro armónico del flujo en la rama magnetizante y su porcentaje de $THD = 3.99\,\%$. La distorsión en las formas de onda se deben a la presencia de las ramas magnetizantes del transformador, cuyo punto de operación se encuentra en la región de saturación.



(a) Forma de onda del voltaje en el nodo 9



(b) Flujo en la rama magnetizante 1

Figura 4.9: Formas de onda de algunas variables de estado y su contenido armónico

La Figura 4.10 muestra el comportamiento de la eficiencia relativa obtenida con la plataforma de OpenMP, con el uso de 1 a 8 procesadores.

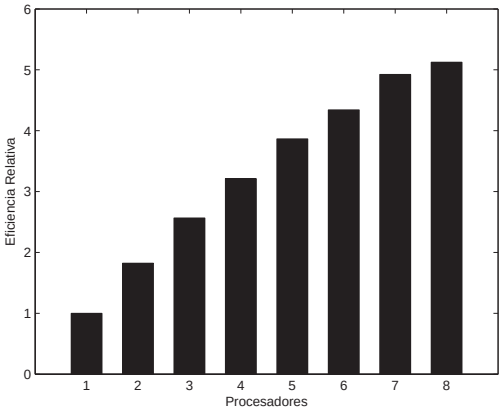


Figura 4.10: Eficiencia relativa con el uso de 1-8 procesadores para el Caso de Estudio 4.5

De la Figura 4.10 se observa que la Eficiencia Relativa máxima es cercana a 5.0 y se obtiene con 8 procesadores, que es la cantidad máxima de procesadores con los que se disponen.

4.6. Sistema de IEEE de 118 nodos

El sistema eléctrico de prueba para éste caso de estudio está formado por 186 líneas de transmisión, 118 bancos de capacitores, 54 generadores y 9 ramas magnetizantes, conectadas a los nodos 8, 26, 30 38, 63, 64, 65, 68 y 81. La dinámica del sistema presentado en este caso de estudio es representado por medio de 367 EDO's.

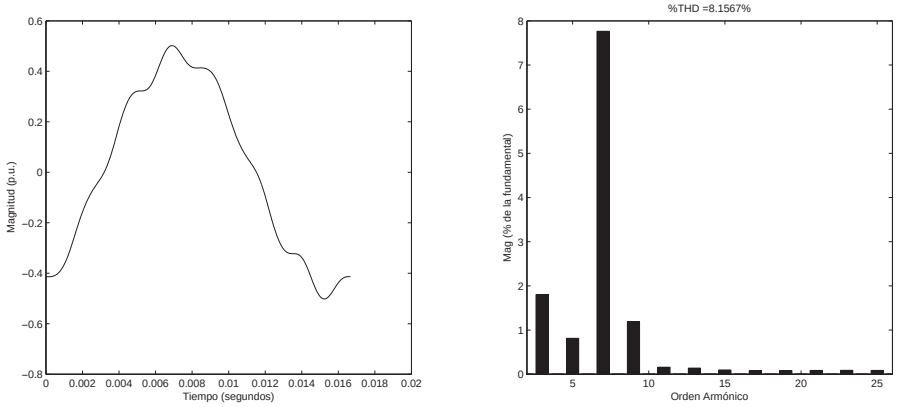
La Tabla 4.5 muestra el número aplicaciones requeridos por los métodos de DN y MED para determinar la solución EEP.

Tabla 4.5: Convergencia de los métodos DN y MED

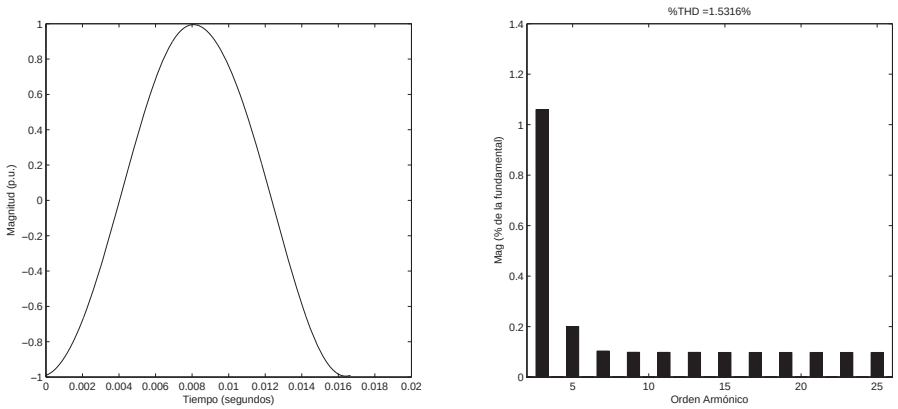
Iteraciones	DN	MED
1	4.708612E-02	2.441780E-04
2	1.422345E-04	6.534880E-06
3	1.596124E-10	2.303364E-06
4		4.935502E-07
5		8.341426E-08

De la Tabla 4.5 se puede apreciar que el método de diferenciación númerica requiere de 4 iteraciones (1108 ciclos) para determinar la solución en EEP, mientras que el método de la MED requiere de 5 iteraciones (9 ciclos) para satisfacer el criterio de convergencia pre-especificado. El método convencional que es FB requiere de 1895 periodos completos de integración.

En la Figura 4.11 se muestra la evolución en el tiempo de algunas variables de estado representativas de la solución en EEP. La Figura 4.11a muestra la forma de onda de la corriente que circula por la línea de transmisión conectada entre el nodo 64 al nodo 65; el contenido armónico de esta forma de onda está asociado principalmente con el séptimo armónico con magnitud cercana al 8 % y tiene un porcentaje THD de 8.15 % . En la Figura 4.11b se presenta el flujo que circula por la rama magnetizante conectada al nodo 8 y su contenido armónico.



(a) Forma de onda de la corriente en la Línea 64-65



(b) Flujo en la rama magnetizante 1

Figura 4.11: Formas de onda de algunas variables de estado y su contenido armónico

La eficiencia relativa obtenida para este caso de estudio se presenta en la Figura 4.12. Obsérvese que la eficiencia relativa máxima que fue obtenida para este caso es cercana a 7.0. Para obtener esta eficiencia relativa es necesario utilizar 8 procesadores.

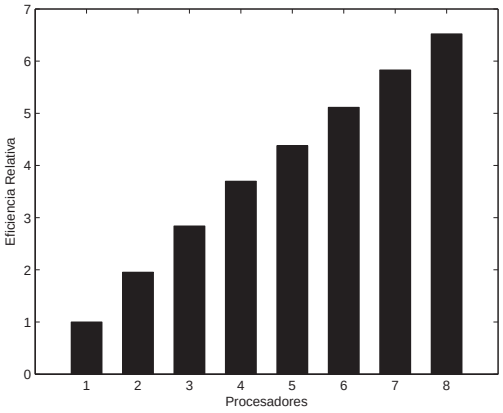


Figura 4.12: Eficiencia relativa con el uso de 1-8 procesadores para el Caso de Estudio 4.5

4.7. Conclusiones

El presente capítulo ha mostrado el uso del método de la Matriz Exponencial Discreta desarrollado e implementado con el objeto de realizar un análisis eficiente de la solución en EEP de redes eléctricas monofásicas de pequeña y mediana escala, en el dominio del tiempo.

Se ha mostrado el potencial de las técnicas de procesamiento en paralelo basado en la plataforma de OpenMP aplicadas a la determinación del EEP de redes eléctricas con componentes no lineales y variantes en el tiempo. Se demostró que conforme el sistema de ecuaciones que describen el comportamiento del número eléctrico analizado, se obtienen incrementos mayores en la eficiencia relativa con el uso de varios procesadores. Por ejemplo, para los casos de estudio analizados, se llega a obtener una eficiencia relativa de 7 con los 8 procesadores. Se esperaría que esta se incrementará con un número mayor de procesadores activos.

Capítulo 5

Conclusiones Generales y Sugerencias para Trabajo Futuro

5.1. Conclusiones generales

En esta tesis se ha aplicado una metodología eficiente para la determinación de la solución en Estado Estacionario Periódico de redes eléctricas con componentes no lineales. Es un método Newton basado en la Matriz Exponencial Discreta, se empleó una extensión en series de Taylor en conjunto con el método de Escalamiento y Cuadratura para aproximar las matrices exponenciales necesarias para el proceso de solución.

Se incorporó de manera satisfactoria la multiplicación en forma dispersa al método de la Matriz Exponencial Discreta para aprovechar que el Jacobiano durante la expansión en series de Taylor mantiene su dispersidad. Uno de los procesos que demandan más tiempo de computo que está relacionado con el método de la Matriz Exponencial Discreta, es el que esta involucrado con el escalamiento de la matriz resultante de la expansión en series de Taylor. Para reducir el tiempo de computo se propuso un esquema en el que la potencia de la matriz se realiza como una sucesión de multiplicaciones matriciales paralelizadas y se obtienen mejoras en la eficiencia relativa del método de la Matriz Exponencial Discreta. Se mostró en los casos de estudio que conforme se aumenta el número de EDO's que describen la dinámica del sistema analizado, se obtienen mejores eficiencias relativas.

El método de la Matriz Exponencial Discreta requiere de un paso de integración muy pequeño para que el método tenga una convergencia cuadrática o muy cercana a ella, ya que este método está basado en la aproximación de Φ por medio de matrices exponenciales, permite el cálculo de Φ con sólo la integración de un ciclo inicial.

5.2. Recomendaciones para trabajos futuros

Tomando como referencia la investigación realizada en esta tesis se proponen a manera de trabajos futuros:

- Paralelizar el cálculo de las matrices exponenciales con un esquema de procesamiento en paralelo basado en multicomputadoras.
- Con el objeto de hacer una comparación alternativa entre los métodos en el dominio del tiempo para la solución en EEP de redes eléctricas no lineales, hacer un análisis de operaciones que realiza cada método para la determinación de la solución en EEP y en base a este análisis realizar una comparación.
- Implementar el MED con procesamiento en paralelo y dispersidad, métodos de integración de error controlado (paso variable).

Apéndice A

Runge Kutta de cuarto orden.

El proceso de integración numérico basado en el método de Runge-Kutta de cuarto orden (RK4) utilizado para los casos, la solución de 2.1 por medio del método de Runge-Kutta de cuarto orden está dada por:

$$x_{n+1} = x_n + \frac{1}{6}(k_1 + 2k_2 + 2_3 + k_4)$$

en donde:

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1)$$

$$k_3 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2)$$

$$k_4 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_3)$$

en donde x_n representa el vector de condiciones iniciales de las variables de estado y h representa el paso de integración. Sin embargo se puede utilizar cualquier otro método de integración.

Apéndice B

Ecuaciones: Red de 2 nodos y 1 STATCOM

El sistema de EDO's que describe la dinámica del STATCOM se presentan en [Segundo 2009], a continuación se presentan las EDO's que modelan la red de dos nodos es del tipo $\dot{x} = f(t, x)$, donde:

$$x = \begin{pmatrix} i_{\mathrm{s1a}} \\ i_{\mathrm{s1b}} \\ i_{\mathrm{s1c}} \\ i_{\mathrm{stata}} \\ i_{\mathrm{statb}} \\ i_{\mathrm{statc}} \\ i_{\mathrm{L1a}} \\ i_{\mathrm{L1c}} \\ i_{\mathrm{L1b}} \\ i_{\mathrm{L2a}} \\ i_{\mathrm{L2c}} \\ i_{\mathrm{L2b}} \\ V_{\mathrm{dc}} \\ V_{\mathrm{d1}} \\ V_{\mathrm{dm}} \\ m_{\mathrm{Ei}} \\ \alpha_{\mathrm{Ei}} \end{pmatrix} ; f(t,x) = \begin{pmatrix} \frac{-R_{\mathrm{s1}} i_{\mathrm{s1a}} - V_{1a} + V_{\mathrm{s1a}}}{L_{\mathrm{s1}}} \\ \frac{-R_{\mathrm{s1}} i_{\mathrm{s1b}} - V_{1b} + V_{\mathrm{s1b}}}{L_{\mathrm{s1}}} \\ \frac{-R_{\mathrm{s1}} i_{\mathrm{s1c}} - V_{1c} + V_{\mathrm{s1c}}}{L_{\mathrm{s1}}} \\ \frac{-R_{\mathrm{stat}} i_{\mathrm{stata}} - V_{1a} + V_{\mathrm{stata}}}{L_{\mathrm{stat}}} \\ \frac{-R_{\mathrm{stat}} i_{\mathrm{statb}} - V_{1b} + V_{\mathrm{statb}}}{L_{\mathrm{stat}}} \\ \frac{-R_{\mathrm{stat}} i_{\mathrm{statc}} - V_{1c} + V_{\mathrm{statc}}}{L_{\mathrm{stat}}} \\ \frac{-R_{\mathrm{L1}} i_{\mathrm{L1a}} - V_{1a} + V_{\mathrm{s2a}}}{L_{\mathrm{L1}}} \\ \frac{-R_{\mathrm{L1}} i_{\mathrm{L1b}} - V_{1b} + V_{\mathrm{s2b}}}{L_{\mathrm{L1}}} \\ \frac{-R_{\mathrm{L1}} i_{\mathrm{L1c}} - V_{1c} + V_{\mathrm{s2c}}}{L_{\mathrm{L1}}} \\ \frac{-R_{\mathrm{L1}} i_{\mathrm{L1a}} - V_{1a} + V_{\mathrm{s2a}}}{L_{\mathrm{L1}}} \\ \frac{-R_{\mathrm{L1}} i_{\mathrm{L1b}} - V_{1b} + V_{\mathrm{s2b}}}{L_{\mathrm{L1}}} \\ \frac{-R_{\mathrm{L1}} i_{\mathrm{L1c}} - V_{1c} + V_{\mathrm{s2c}}}{L_{\mathrm{L1}}} \\ \frac{1}{C_{\mathrm{dc}}} \begin{pmatrix} S_{\mathrm{Ea}} & S_{\mathrm{Eb}} & S_{\mathrm{Ec}} \end{pmatrix} i_{\mathrm{stat}} \\ -888.5765 V_{\mathrm{d1}} - 628.3185 V_{\mathrm{dm}} + 62.8318 V_d \\ 628.3185 V_{\mathrm{d1}} \\ K_{\mathrm{mi}} \left(V_{\mathrm{tref}} - V_{\mathrm{dm}} \right) \\ K_{\alpha \mathrm{i}} \left((V_{\mathrm{dc}})^{\,2} - (V_{\mathrm{dcref}})^{\,2} \right) \end{pmatrix}$$

Parámetros Eléctricos

$R_{\mathrm{s1}} = 0.1\Omega$
$L_{\mathrm{s1}} = 0.001H$
$R_{\mathrm{L1}} = 0.015H$
$R_{\mathrm{L2}} = 0.15\Omega$
$L_{\mathrm{L2}} = 0.01H$
$C_{\mathrm{dc}} = 1000\mu\mathrm{F}$
$R_{\mathrm{stat}} = 0.05\Omega$
$r_s = 1\mathrm{m}\Omega$

Ganancias de los controladores

$K_{\mathrm{mp}} = 0.004$
$K_{\mathrm{mi}} = 1.6$
$K_{\alpha\mathrm{p}} = 1.6 \times 10^{-6}$
$K_{\alpha\mathrm{i}} = 6.0 \times 10^{-5}$

Voltajes de referencia

$V_{\mathrm{dcref}} = 500\mathrm{Volts}$
$V_{\mathrm{1ref}} = 200\sqrt{\frac{2}{3}}\mathrm{Volts}$

Fuentes de voltaje

$$V_{\mathrm{s1}} = \begin{pmatrix} V_{\mathrm{s1a}} \\ V_{\mathrm{s1b}} \\ V_{\mathrm{s1c}} \end{pmatrix} = 200\sqrt{\tfrac{2}{3}} \begin{pmatrix} \cos(2\pi\mathrm{ft}) \\ \cos\left(2\pi\mathrm{ft} - \frac{2\pi}{3}\right) \\ \cos\left(2\pi\mathrm{ft} - \frac{2\pi}{3}\right) \end{pmatrix} Volts$$

$$V_{\mathrm{s2}} \begin{pmatrix} V_{\mathrm{s2a}} \\ V_{\mathrm{s2b}} \\ V_{\mathrm{s2c}} \end{pmatrix} = 200\sqrt{\tfrac{2}{3}} \begin{pmatrix} \cos(2\pi\mathrm{ft} - 0.4363323129985824) \\ \cos\left(2\pi\mathrm{ft} - \frac{2\pi}{3} - 0.4363323129985824\right) \\ \cos\left(2\pi\mathrm{ft} - \frac{2\pi}{3} - 0.4363323129985824\right) \end{pmatrix} \mathrm{Volts}$$

Ecuación del control PI para el índice de modulación de amplitud m_E

$$m_E = K_{\mathrm{mp}}\left(V_{\mathrm{1ref}} - V_{\mathrm{dm}}\right) + m_{\mathrm{Ei}}$$

Ecuación del control de PI del ángulo del voltaje en terminales del STATCOM α_E

$$\alpha_E = K_{\alpha}\left(\left(V_{\mathrm{dc}}\right)^2 - \left(V_{\mathrm{dcref}}\right)^2\right) + \alpha_{\mathrm{Ei}}$$

Índice de modulación de frecuencia

$m_f = 15$

Generación de la seña triangular

$$\mathrm{tri} = \tfrac{2}{\pi}\mathrm{asin}\left(\sin\left(2\pi\mathrm{m}_f\mathrm{ft} + \pi\right)\right)$$

Generación de las señales de control para el proceso de modulación PWM
$V_{\mathrm{EA}} = m_E \cos(\omega \mathrm{t} + \alpha_E)$ $V_{\mathrm{EB}} = m_E \cos\left(\omega \mathrm{t} + \alpha_E - \frac{2\pi}{3}\right)$ $V_{\mathrm{Ec}} = m_E \cos\left(\omega \mathrm{t} + \alpha_E + \frac{2\pi}{3}\right)$

Genaración de las funciones de conmutación
$S_{\mathrm{Ea}} = \frac{(\tanh(25(V_{\mathrm{EA}} - \mathrm{tri}))+1)}{2}$ $S_{\mathrm{Eb}} = \frac{(\tanh(25(V_{\mathrm{EB}} - \mathrm{tri}))+1)}{2}$ $S_{\mathrm{Ec}} = \frac{(\tanh(25(V_{\mathrm{EC}} - \mathrm{tri}))+1)}{2}$

Voltaje en terminales del STATCOM
$\begin{pmatrix} V_{\mathrm{stata}} \\ V_{\mathrm{statb}} \\ V_{\mathrm{statc}} \end{pmatrix} = \begin{pmatrix} i_{\mathrm{stata}} \\ i_{\mathrm{statb}} \\ i_{\mathrm{statc}} \end{pmatrix} r_s + V_{\mathrm{dc}} \begin{pmatrix} S_{\mathrm{Ea}} \\ S_{\mathrm{Eb}} \\ S_{\mathrm{Ec}} \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \frac{V_{\mathrm{dc}}}{3} (S_{\mathrm{Ea}} + S_{\mathrm{Eb}} + S_{\mathrm{Ec}})$

Ecuación del voltaje V_1
$V_1 = \frac{1}{1+\frac{L_{\mathrm{s1}}}{L_{\mathrm{L1}}}+\frac{L_{\mathrm{s1}}}{L_{\mathrm{L2}}}+\frac{L_{\mathrm{s1}}}{L_{\mathrm{stat}}}} \left(V_{\mathrm{s1}} + L_{\mathrm{s1}} \left(\frac{V_{\mathrm{s2}}+R_{\mathrm{L1}} i_{\mathrm{stat}}}{L_{\mathrm{L1}}} + \frac{V_{\mathrm{s2}}+R_{\mathrm{L1}} i_{\mathrm{L2}}}{L_{\mathrm{L1}}} + \frac{V_{\mathrm{stat}}+R_{\mathrm{stat}} i_{\mathrm{stat}}}{L_{\mathrm{stat}}}\right) - R_{\mathrm{s1}} (i_{\mathrm{L1}} + i_{\mathrm{L2}} + i_{\mathrm{stat}})\right)$

Apéndice C

Modelado de los elementos monofásicos

C.1. Linea de transmisión

El modelo de la línea de transmisión, utilizado en este trabajo, está representada por una resistencia R conectada en serie con un inductor L, tal como se ilustra en la Figura C.1.

Figura C.1: Modelo de la línea de transmisión

La aplicación de una LVK en el circuito que se muestra en la Figura C.1, obteniéndose:

$$\frac{di}{dt} = \frac{V_A - V_B}{L} - \frac{\mathrm{iR}}{L} \tag{C.1}$$

C.2. Generador

El generador es representado por medio de una fuente sinusoidal de voltaje constante, conectada a un inductor, se muestra en la Figura C.2 el modelo del generador.

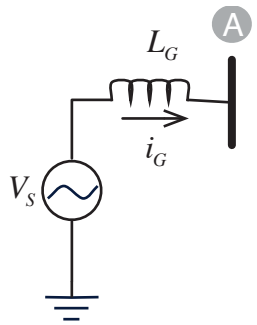


Figura C.2: Modelo del generador

Aplicando LVK al circuito de la Figura C.2, y despejando di_G se obtiene:

$$\frac{di_G}{dt} = \frac{V_S - V_A}{L_G} \tag{C.2}$$

El voltaje en terminales del generador está determinado por la Ecuación (C.3)

$$V_S = M\text{sen}(\omega\text{t} + \phi) \tag{C.3}$$

donde:

M : Valor pico del voltaje.

ϕ : Ángulo de fase del voltaje expresado en radianes.

ω : Velocidad angular.

C.3. Banco de capacitores

La Figura (C.3) muestra un banco de capacitores monofásico conectado a un nodo del sistema. En el modelado se considera que todas las corrientes que incien en el nodo, así como todas las corrientes que salen del nodo, a excepción de la corriente en el capacitor.

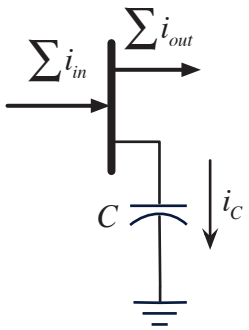


Figura C.3: Modelo del banco de capacitores

donde:

$\sum i_{in}$: Suma de corrientes que inciden en el nodo.

$\sum i_{out}$: Suma de corrientes que salen del nodo.

Aplicando LCK en el nodo se tiene,

$$\sum i_{in} = \sum i_{out} + i_C \tag{C.4}$$

por lo tanto, el voltaje en el nodo se describe con la ecuación:

$$\frac{dV_C}{dt} = \frac{\sum i_{in} - \sum i_{out}}{C} \tag{C.5}$$

C.4. Rama magnetizante

La Figura C.4 muestra la conexión en el nodo A de una rama magnetizante.

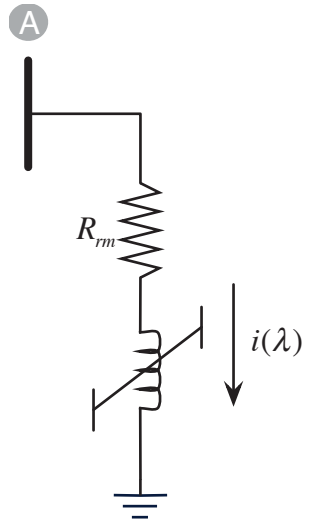


Figura C.4: Modelo de la rama magnetizante

Por medio de la aplicación de una LVK alrededor de la malla da como resultado la Ecuación (C.6)

$$V_A = V_R + V_L \tag{C.6}$$

donde:

$$V_R = R_{rm} i\,(\lambda) \tag{C.7}$$

$$V_L = \frac{d\lambda}{dt} \tag{C.8}$$

Sustituyendo (C.7) y (C.8) en (C.6) y despejando $\frac{d\lambda}{dt}$ se tiene que:

$$\frac{d\lambda}{dt} = V_A - R_{rm} i(\lambda)$$

El efecto de la saturación en la rama magnetizante es representado mediante una aproximación basada en un polinomio de grado n. El comportamiento no lineal de la rama magnetizante esta expresado por medio de la relación no lineal:

$$i(\lambda) = \lambda^n$$

donde:

n : Tiene un valor de 5.

C.5. Horno de arco eléctrico

Considere la conexión de un horno de arco en el nodo A del sistema, tal y como se ilustra en la Figura C.5.

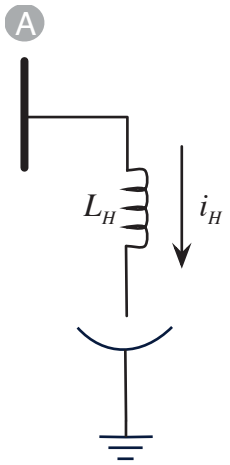


Figura C.5: Modelo del horno de arco eléctrico

El modelo presentado en la C.5 esta basado en el principio de conservación de la energía, representado por la ecuación:

$$p_1 + p_2 = p_3 \tag{C.9}$$

donde:

$$p_1 = k_1 r^n \tag{C.10}$$

$$p_2 = k_2 r \frac{dr}{dt} \tag{C.11}$$

$$p_3 = \frac{\frac{k_3}{r^m}}{r^2} i^2 \tag{C.12}$$

Sustituyendo las Ecuaciones (C.10),(C.11) y (C.12) en (C.9), se encuentra la ecuación diferencial del arco:

$$k_1 r^n + k_2 r \frac{dr}{dt} = k_3 r^{-(m+2)} i^2 \tag{C.13}$$

El voltaje del arco está dado por:

$$v = k_3 r^{-(m+2)} i^2 \tag{C.14}$$

Por medio de la aplicación de una LVK alrededor de la malla formada de la Figura C.5, se tiene que:

$$v_A = V_{LH} + V_{RH} \tag{C.15}$$

donde:

$$V_{LH} = L_H \frac{di_H}{dt} \tag{C.16}$$

$$V_{RH} = k_3 r^{-(m+2)} i_H \tag{C.17}$$

sustityendo y despejando el término $\frac{di_H}{dt}$ se tiene que:

$$\frac{di_H}{dt} = \frac{-k_3 r^{-(m+2)} i_H + v_A}{L_H} \tag{C.18}$$

la derivada del radio del arco está dada por:

$$\frac{dr}{dt} = \frac{k_3}{k_2} r^{-(m+3)} i_H^2 - \frac{k_1}{k_2} r^{n-1} \tag{C.19}$$

Apéndice D

Archivos de datos para generación de EDO's

En este Apéndice se presenta los archivos de entrada que requiere el software de [Ramos 2007] para la generación de las EDO's que describen el comportamiento de los sistemas analizados en los casos de estudio.

Tabla D.1: Archivo de entrada para la simulación del caso de estudio 4.2

3 9					
L_1	1	2	0.01	0.1	
L_2	1	3	0.01	0.1	
L_3	2	3	0.01	0.1	
C_1	1	0	0.01		
C_2	2	0	0.01		
C_3	3	0	0.01		
RM_1	2	0	0.1	0.1	5
RM_2	3	0	0.1	0.1	5
G_1	1	0	1	0	0.001

Tabla D.2: Archivo de entrada para la simulación del caso de estudio 4.3

14	38				
L_1	1	2	0.02	0.06	
L_2	1	5	0.05	0.22	
L_3	2	3	0.05	0.2	
L_4	2	4	0.06	0.18	
L_5	2	5	0.06	0.17	
L_6	3	4	0.07	0.17	
L_7	4	5	0.01	0.04	
L_8	4	7	0.0001	0.21	
L_9	4	9	0.0001	0.56	
L_10	5	6	0.0001	0.25	
L_11	6	11	0.1	0.2	
L_12	6	12	0.12	0.26	
L_13	6	13	0.07	0.13	
L_14	7	8	0.0001	0.18	
L_15	7	9	0.0001	0.11	
L_16	9	10	0.03	0.09	
L_17	9	14	0.13	0.27	
L_18	10	11	0.08	0.2	
L_19	12	13	0.22	0.2	
L_20	13	14	0.17	0.35	
C_1	1	0	0.1		
C_2	2	0	0.1		
C_3	3	0	0.1		
C_4	4	0	0.1		
C_5	5	0	0.1		
C_6	6	0	0.1		
C_7	7	0	0.1		
C_8	8	0	0.1		
C_9	9	0	0.1		
C_10	10	0	0.1		
C_11	11	0	0.1		
C_12	12	0	0.1		
C_13	13	0	0.1		
C_14	14	0	0.1		
RM_1	12	0	0.1	0.1	5
RM_2	13	0	0.1	0.1	5
G_1	1	0	1	0	0.001
G_2	2	0	1	0	0.001

Apéndice E

Biblioteca CSparse

E.0.1. Instalación de CSparse

Para utilizar la librería de "cs.h" se deberá instalar el paquete CSparse que se puede descargar de [cise.ufl.edu], para instarlo solamente se tiene que acceder a el directorio CSparse y ejecutar el comando Makefile. Para ejecutarlo, solamente se deberá teclear en la línea de comandos make. El sistema operativo automáticamente busca un archivo con el nombre Makefile, por lo tanto, si se tiene un archivo con el nombre Makefile y se teclea make en la línea de comandos, el archivo Makefile del directorio actual será ejecutado. Se puede anular esta búsqueda con el comando make -f makefile. Para ejecutar un programa con la librería "cs.h", se debe de ligar la biblioteca, así como los directorios asociados con la cs.h, entonces para ejecutar un programa se debe teclear:

```
gcc -0 -I/home/.../CSparse/Include -o programa programa.c /home/.../CSparse/Lib/libcsparse.a
```

E.0.2. Funciones de utilería

Se requieren algunas funciones de utilería para crear esta estructura de datos. La Tabla E.1 muestra las funciones cs_malloc, cs_calloc, cs_realloc y cs_free, que son paquetes de las funciones de manejo de memoria equivalentes en C.

Tabla E.1: Funciones de utilería de "cs.h" [Timothy 2006]

cs_realloc	Cambia el tamaño de un bloque de memoria.
cs_spalloc	Crea una matriz dispersa de $n \times n$ que puede contener hasta $nzmax$ entradas
cs_spfree	Libera una matriz dispersa de la forma cs
cs_sprealloc	Cambia el número máximo de entradas $nzmax$ que la matriz cs puede contener

E.0.3. Forma triple

Las funciones de utilería pueden ubicar espacio para una matriz dispersa, pero sin definir su contenido. La manera más simple de construir una matriz cs es primero ubicar una matriz de forma triple. Las aplicaciones normalmente crean una matriz de ésta manera, en lugar de definirlas de forma estática como se muestra en la Tabla E.2:

Tabla E.2: Construcción de la estructura cs en forma estática

```
cs *T ;
int *Ti, *Tj ;
double *Tx ;
T = sc_spalloc (m, n, nz, 1, 1) ;
Ti = T->i ; Tj = T->p ; Tx = T->x ;
```

Enseguida, coloca cada entrada de la matriz dispersa en los arreglos Ti, Tj y Tx. La k-ésima entrada tiene un índice renglón $i = Ti[k]$, un índice columna $j = Tj[k]$, y un valor numérico $a_{ij} = Tx[k]$. Las entradas pueden aparecer en orden arbitrario. Establezca $T->nz$ para que sea el número de entradas en la matriz.

La función cs_entry es útil si el número de entradas en la matriz no se conoce cuando la matriz se ubica por primera vez. Si el espacio no es suficiente para la siguiente entrada, el tamaño de los arreglos $T->i$, $T->j$ y $T->x$ se duplican. Las dimensiones de T se incrementan según se necesiten.

La función cs_compress convierte esta forma triple T a una forma de matriz de columna condensada C. Primero, se reubican C y un espacio de trabajo tamaño n. Enseguida, se calcula el número de entradas en cada columna C, y el arreglo de apuntadores columna Cp se construye como la forma cumulativa de los conteos de columna. Los conteos en w también se cambian por una copia de Cp. cs_compress itera a través de cada entrada en la matriz triple. El apuntador columna $w[Tj[k]]$ se encuentra e incrementa posteriormente. Esto determina la ubicación p donde el índice de renglón $Ti[k]$ y el valor numérico $Tx[k]$ se colocan en C. Finalmente se libera el espacio de trabajo y el resultado C se regresa en forma de columna condensada.

En la Tabla E.3 se presentan las funciones y definiciones de CSparse.

Tabla E.3: Tabla de funciones de la librería CSparse

cs: Matriz dispersa en forma triple o columna condensada

```
typedef struct cs_sparse
 {
    int     nzmax;  /* Máximo número de entradas */
    int     m;      /* Número de filas */
    int     n;      /* Número de columnas */
    int     *p;     /* Punteros de columna (n+1) o índices de columna (nzmax) */
    int     *i;     /* Índices de filas (nzmax) */
    double  *x;     /* Valores diferentes de cero (nzmax) */
    int     nz;     /* #de entradas en matriz triple, -1 para columna condensada */
    } cs;
```

cs_add: $C = \alpha A + \beta B$

```
cs *cs_add(const cs *A, const cs *B, double alpha, double beta);
```

Suma dos matrices dispersa.

A	entrada	Matriz dispersa
B	entrada	Matriz dispersa
alpha	entrada	Escalar
beta	entrada	Escalar
	retorna	$C = alpha * A + beta * B$ o en caso de error Null

cs_compress: Convierte de forma triple a columna condensada

```
cs *cs_compress (const cs *T) ;
```

Convierte un matriz de la forma triple a una matriz de la forma columna condensada C. Las columnas de C no están ordenadas, y puede haber entradas duplicadas en C.

T	entrada	Matriz en forma triple.
	retorna	C si es correcto o Null en caso de error

cs_dupl: Remueve entradas duplicadas

```
int cs_dupl (cs *A) ;
```

Remueve y suma entradas duplicadas de una matriz dispersa.

A	entrada/salida	Matriz en forma triple.
	retorna	1 si es correcto o 0 en caso de error

cs_entry: Añade entradas a una matriz en forma triple

int cs_entry (cs *T, int i, int j, double x) ;

La dimension de T y el espacio en la memoria se aumentan si es necesario.

T	entrada/salida	Matrix triple; añade nueva entrada en la salida.
	i	índice de la fila de la nueva entrada.
	j	índice de la columna de la nueva entrada.
	x	Valor de la nueva entrada.
	retorna	1 si es correcto o 0 en caso de error.

cs_multiply: $C = AB$

int *cs_multiply(const cs *A, const cs *B);

Multiplicación de matrices dispersas, $C = AB$.

A	entrada	Matriz dispersa.
B	entrada	Matriz dispersa
y	entreda/salida	m
	retorna	1 si es correcto o 0 en caso de error.

cs_norm: Norma-1

double cs_norm(const cs *A);

Calcula la norma-1 de una matriz dispersa.

A	entrada	Matriz dispersa.
	retorna	Norma-1 si es correcto o 0 en caso de error.

Bibliografía

[Almasi y Gotlieb 1994] G.S. Almasi y A. Gotlieb. "Highly Parallel Computing". Benjamin Cummings, New York. 1994.

[Alvarado *et al*. 1992] Alvarado F. *et al*. "Parallel Processing in Power Systems Computation", IEEE Transactions on Power Systems, Vol. 7, No. 2, págs. 629-637, Mayo 1992.

[Aprille y Trick 1972] T. J. Aprille Jr. y T. N. Trick, "Steady State Analysis of Nonlinear Circuits with Periodic Inputs", Proc. IEEE, vol. 60, no. 1, págs. 108-114, Enero 1972.

[Armanazi 1973] Armanazi A. N., "Steady-State of Piecewise-Linear Systems with periodic Inputs", Proc. IEEE, vol. 61, págs. 789-790, Junio 1973.

[Arrillaga y Medina 1995] Arrillaga J., Medina A., Lisboa M., Cavia M., Sanchez P., "The Harmonic Domain. A Frame of Reference for Power System Harmonic Analysis". IEEE Transactions on Power Systems, Vol. 10, No. 1, págs. 433-440, Febrero 1995.

[Banerjee y Verghese 2002] S. Banerjee y G. C. Verghese, "Nonlinear Phenomena in Power Electronics, Attractors, Bifurcations, Chaos and Nonlinear Control" IEEE Press, 2002.

[Bedrosian y Vlach 1992] D. Bedrosian and J. Vlach, "An Accelerated Steady State Method for Networks With Internally Controlled Switches", IEEE Trans. Circ. And Syst., Part I, vol. 16, no. 7, págs. 520-530, Julio 1992.

[Callaghan y Arrillaga 1990] C. D. Callaghan and J. Arrillaga, "Convergence criteria for iterative harmonic analysis and its application to static convertors," Proc. IEEE/ICHPS IV International Conference on Harmonics in Power Systems, Budapest, Hungary, págs. 38-43, Octubre 1990.

[Chandra *et al* 2001] R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald and R. Menon, "Parallel Programming in OpenMp" Morgan Kaufmann Publishers, Academic Press, San Francisico, USA, 2001.

[cise.ufl.edu] http://www.cise.ufl.edu/research/sparse/CSparse/

[D'Angelo 1970] Henry D'Angelo, "Linear Time Varying Systems: Analysis and Synthesis", Allyn and Bacon, Boston, 1970.

[Donde y Hiskens 2006] V. Donde y I. A. Hiskens, "Shooting Method for Locating Grazing Phenomena in Hybrid Systems", International Journal of Bifurcation and Chaos, vol 16, no. 3, págs. 671-692, Marzo 2006.

[ee.washington.edu] http://www.ee.washington.edu/research/pstca/

[Flynn 1972] M.J. Flynn. Some computer organizations and their effectiveness. IEEE Transactions on Computers, 1 págs. 948-960, 1972.

[Foster 1994] Foster I., "Designing and Building Parallel Programs", Addison Wesley, 1994.

[Geist *et al*. 1994] Geist A., Beguelin A. y Dongarra J., "PVM: Parallel Virtual Machine", MIT Press, 1994.

[Howroyd 1982] D. C. Howroyd, "Case studies in distortion on the public supply system," IEE Conference on Sources and Effects of Power System Disturbances, no. 210, págs. 215-220, 1982.

[J.Usaola-Garcia 1990] J. Usaola-Garcia, "Steady state in power systems with nonlinear elements through a hybrid procedure of analysis in the time and frequency domains", Ph.D. dissertation (in Spanish), Universidad Politécnica de Madrid, 1990.

[Jin 1994] Jin L., "Parallel Processing: Exploring the Archictectures and Algorithms Close Relationship", IEEE POTENTIALS, págs. 17-20, Diciembre 1994-Enero 1995.

[Kundert *et al*. 1990] Kundert K.S., White J.K. y Sangiovanni-Vincentelli A. "Steady-State Methods for Simulating Analog and Microwave Circuits", Kluver Academic Publishers, Estados Unidos de América, 1990.

[Kleiman *et al*. 1992] Kleiman S., Smaalders B., Stein D., Shah D., "Writing Multithreading Code in Solaris", 1992.

[Lewis y Berg 1998] Lewis B. y Berg D.J., "Multithreaded Programming with Pthreads", Prentice Hall, 1998.

[Li y Tymerski 2000] Li y R. Tymerski, "Comparison of Simulation Algorithms for Accelerated Determination of Periodic State of Switched Networks", IEEE Trans. on Ind. Electronics, vol. 47, no. 6, págs.1278-1285, 2000.

[Mahmoud y Schultz 1982] A. A. Mahmoud and R. D. Schultz, "A method for analyzing harmonic distribution in a.c. power systems," IEEE Trans. on Power Apparatus and Systems, vol. PAS-101, no. 6, págs. 1815-1824, 1982.

[Medina *et al*. 2007] A Medina, N R Watson, P F Ribeiro, and C J Hatziadoniu, Chapter 5. Harmonic analysis in frequency and time domains, tutorial course on harmonics modeling and simulation, IEEE PES, 2007.

[Medina *et al*. 2006] Aurelio Medina, Antonio Ramos-Paz, Claudio Rubén Fuerte-Esquivel, "Efficient computation of the periodic steady state solution of nonlinear electric systems applying parallel processing techniques", COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering, Vol. 25 Iss: 4, págs. 900-915, 2006.

[Medina y Arrillaga 1990] A. Medina, J. Arrillaga, "Sparsity-oriented hybrid formulation of linear multiports and its application to harmonic analysis," IEEE Trans. on Power Delivery, vol. 5, no. 3, págs. 1453-1458, July 1990.

[Medina y Ramos 2005a] Medina A., Ramos-Paz A., "Newton Techniques for the Steady State Solution of Nonlinear Electric Networks", WSEAS Transactions on Circuits and Systems. No. 8, Vol. 4, 842-849, Agosto 2005.

[Medina y Ramos 2005b] Medina A., Ramos-Paz A., "PVM and MT Parallel Processing Platforms Applied to the Computation of Driving Point Impedances in Power Systems", WSEAS Transactions on Circuits and Systems. No. 11, Vol. 4, págs. 1702-1709, Noviembre 2005.

[Moler y Loan 2003] Moler y C. V. Loan, "Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later", Society for Industrial and Applied Mathematics, vol. 45, no. 1, págs. 1-46, March 2003.

[Nakhla y Vlach 1976] Nakhla M. S. y Vlach J., "A Piecewise Harmonic Balance Technique for Determination of Periodic Response of Nonlinear Systems". IEEE Transactions on Circuits and Systems, Vol. CAS-23, No. 2, págs. 85 – 91, Febrero 1976.

[Nayfeh 1995] H. Nayfeh, "Applied Nonlinear Dynamics: Analytical, Computational, and Experimental Methods", John Wiley & Sons, New York, 1995.

[Nocedal 1999] J. Nocedal, "Numerical Optimization", Springer Verlag, New York, 1999.

[openmp.org] http://openmp.org/

[Parker y Chua 1989] T. S. Parker y L. O. Chua, "Practical Numerical Algorithm for Chaotic Systems", Springer-Verlag, New York, 1989.

[Quinn 2004] M. J. Quinn, "Parallel Programming in C with MPI and OpenMP", McGraw- Hill, 1ra edición, New York, USA, 2004.

[Ramos 2002] Ramos Paz A., "Aplicación de Técnicas de Procesamiento en Paralelo a la Solución en Estado Periódico Estacionario de Redes Eléctricas con Componentes no Lineales y Variantes en el Tiempo", Tesis de Maestría, Facultad de Ingeniería Eléctrica, U.M.S.N.H. 2002.

[Ramos 2007] Ramos Paz A. "Técnica para la Generación Automática de Ecuaciones Diferenciales No Autónomas para Representar el Comportamiento Dinámico de Sistemas Eléctricos No- Lineales Incorporando Herramientas Avanzadas de Computo", Tesis de Doctorado, Facultad de Ingeniería Eléctrica, Universidad Michoacana de San Nicolás de Hidalgo, México, 2007.

[Sato y Tinney 1963] N. Sato, W. F. Tinney, "Techniques for Explointing the Sparsity of the Network Admittance Matrix". AIEE Trans. Power Apparatus and Systems, Vol. PAS-82, págs. 944-950, Diciembre 1963.

[Segundo 2009] "Nonlinear Analysis of Power Systems including FACTS and Custom Power Devices Based on Bifurcation Theory and Newton Methods", Tesis de Doctorado, Facultad de Ingeniería Eléctrica, Universidad Michoacana de San Nicolás de Hidalgo, México, 2009.

[Segundo y Medina 2008] Segundo-Ramírez, J., and Medina, A., "Periodic Steady-State Solution of Electric Systems Including UPFCs by Extrapolation to the Limit Cycle." IEEE Transaction on Power Delivery, vol. 23, no. 3, págs. 1506 - 1512, Julio 2008.

[Segundo y Medina 2010a] Segundo-Ramírez, J., and Medina, A., "Computation of the Steady-State Solution of Nonlinear Power Systems by Extrapolation to the Limit Cycle Using a Discrete Exponential Expansion Method." International Journal of Nonlinear Sciences and Numerical Simulation, vol. 11, no. 8, págs. 655-660, Agosto 2010.

[Segundo y Medina 2010b] Segundo-Ramírez, J., and Medina, A., "An Enhanced Process for the Fast Periodic Steady State Solution of Nonlinear Systems by Poincaré Map and Extrapolation to the Limit Cycle." International Journal of Nonlinear Sciences and Numerical Simulation, vol. 11, no. 8, págs. 661-670, Agosto 2010.

[Semlyen y Medina 1995] Semlyen y A. Medina, "Computation of Periodic Steady State in System With Nonlinear Components Using a Hybrid Time and Frequency Domain Methodology," IEEE Trans. Power Syst., vol. 10, no. 3, págs. 1498-1504, Agosto 1995.

[Semlyen y Shlash 2000] A. Semlyen and M. Shlash, "Principles of modular harmonic power flow methodology," Proc. of the IEE, Pt. C, vol. 147, no. 1, págs. 1-6, Enero 2000.

[Subramaniam 1971] Subramaniam P., Malik O.P. "Digital Simulation of a Synchronous Generator in Direct- Phase Quantities". Proc. IEE, Vol. 118, N. 1, págs. 153-160, Enero 1971.

[Timothy 2006] Timothy A. Davis, (2006), "Direct Methods for Sparse Linear Systems", Society for Industrial and Applied Mathematics, Philadelphia, 2006.

[Collins *et. al* 2006] C. Collins, N. Watson y A. Wood, "UPFC modeling in the harmonic domain," IEEE Trans. Power Del., vol. 21, no. 2, págs. 953–1938, Apr. 2006.

[Arrillaga *et.* al 2000] J.Arrillaga, B. C. Smith, N. R. Watson y A. R. Wood, "Power System Harmonic Analysis." New York: Wiley, 2000.

[Chang *et. al* 2006] G. W. Chang, Y. C. Chin y S. H. Lee, "Efficient approach to characterising harmonic currents generated by a cluster of three-phase AC/DC converters," Proc. Inst. Elect. Eng., vol. 153, no. 5, págs. 742–749, Sep. 2006.

[Stavrakakis *et. al* 1990] G. S. Stavrakakis, C. Lefas, A. Pouliezos, "Parallel Processing Computer Implementation of a Real Time DC Motor Drive Fault Detection Algo- rithm", IEE Proceedings, Vol. 137, No. 5, págs. 309-313, Septiembre 1990.

[Mariños *et. al* 1994] Z. A. Mariños, J. L. R. Pereira, Jr. Carneiro, "Fast Harmonic Power Flow Calculation Using Parallel Processing", IEE Proc.-Gener. Transm. Distrib.,. Vol 141, No. 1. Enero 1994, págs. 27-32.

[Garcia *et. al* 2001] N. Garcia, E. Acha, A. Medina, "Swift Time Domain Solutions of Electric Systems Using Parallel Processing", Proceedings of the Sixth International Conference IASTED, Rodas, Grecia, Julio 2001, págs. 172-177.

[Tinney y Walker 1967] Tinney W.F. y Walker J.W., "Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization", Proceedings of IEEE, Vol. 55, págs. 1801-1809, Noviembre 1967.

[Usaola 1990] Usaola-García J. , "Régimen Permanente de Sistemas Eléctricos de Potencia con Elementos No lineales Mediante un Procedimiento Híbrido de Análisis en los Dominios del Tiempo y de la Frecuencia", Tesis de Doctorado, Universidad Politécnica de Madrid, Escuela Técnica Superior de Ingenieros Industriales, 1990.

[Wylie 1951] Wylie C.R., "Advanced Engineering Mathematics", Mc. Graw Hill. Japón, 1951.

[Xia y Heydt 1982] D. Xia and G. T. Heydt, "Harmonic power flow studies, part I – formulation and solution, part II – implementation and practical application," IEEE Trans. on Power Apparatus and Systems, vol. PAS-101, págs. 1257-1270, Junio 1982.

[Gilbert 1994] J. R. Gilbert, "Predicting structure in sparse matrix computations", SIAM J. Matrix Anal. Appl., págs. 62-79, 1994,.

[Lian y Noda 2010] Kuo-Lung Lian y Taku Noda, "A Time-Domain Harmonic Power-Flow Algorithm for Obtaining Nonsinusoidal Steady-State Solutions", IEEE Transactions on Power Delivery, 2010.

[Garcia y Acha 2004] N. Garcia, E. Acha, "Periodic Steady-State Analysis of Large Scale Electric Systems Using Poincare Map and Parallel Processing", IEEE Tran- sactions on Power Systems, Vol. 19, No, 4, págs. 1784-1793, November 2004.